

Blender 3D

Noob to Pro

Contents

0.1	Blender 3D: Noob to Pro	1
0.1.1	About This Book	1
0.1.2	Overview	1
0.1.3	Table of Contents	1
1	Unit 1 - Background	6
1.1	Knowing before Making	6
1.1.1	Additional Resources	7
1.2	What Blender Can Do	7
1.2.1	An Example	7
1.2.2	Steps in the 3D Production Process	7
1.2.3	Additional Resources	8
1.3	3D Geometry	9
1.3.1	Coordinates And Coordinate Systems	9
1.3.2	Axes Of Rotation	10
1.3.3	Additional Resources	11
1.4	Coordinate Transformations	11
1.4.1	Coordinate Transformations	11
1.4.2	Linear Transformations	11
1.4.3	Multiple Transformations	12
1.4.4	Inverse Transformations	12
1.4.5	Projections	13
1.5	Orthographic Views	13
1.5.1	Example	13
1.5.2	Additional Resources	14
1.6	Perspective Views	14
1.6.1	One-point Perspective	14
1.6.2	Two-point Perspective	15
1.6.3	Three-point Perspective	15
1.6.4	Additional Resources	15
1.7	Coordinate Spaces in Blender	15
1.7.1	Global and local coordinates	16
1.7.2	View coordinates	16

1.7.3	Normal coordinates	17
1.7.4	UV Coordinates	17
1.8	Overview	17
1.9	Keystroke, Button, and Menu Notation	18
1.9.1	Hotkeys	18
1.9.2	Mouse Notation	18
1.9.3	Navigating Menus	19
1.9.4	Additional Resources	19
1.10	Non-standard Input Devices	19
1.11	Operating System-specific Issues	20
1.11.1	GNU/Linux	20
1.11.2	Mac OS X	21
1.11.3	Microsoft Windows	21
1.12	Blender User Interface	22
1.12.1	Why Doesn't It Follow UI Conventions For [Insert OS Of Choice Here]?	22
1.12.2	Why Doesn't It Prompt To Save Changes?	23
1.13	Blender Windowing System	23
1.13.1	An Interface Divided	23
1.13.2	Window Headers	24
1.13.3	Window Types	25
1.13.4	The Active Window	25
1.13.5	Resizing Windows	26
1.13.6	Shelves	26
1.13.7	Too Much To Fit	26
1.13.8	Splitting And Joining Windows	26
1.13.9	The Default Workspace	26
1.13.10	Workspace Presets	27
1.13.11	One Document At A Time	27
1.13.12	Scenes	28
1.13.13	Leaving Blender	28
1.13.14	Additional Resources	28
1.14	User Preferences Windows	28
1.14.1	Saving User Preferences	28
1.14.2	Accessing the User Preferences	28
1.14.3	Configuring Your Preferences	29
1.14.4	Additional Resources	30
1.15	Properties Window	30
1.16	The Contexts	30
1.16.1	Render Context	30
1.16.2	Render Layers Context	30
1.16.3	Scene Context	30

1.16.4	World Context	30
1.16.5	Object Context	31
1.16.6	Object Constraints Context	31
1.16.7	Object Modifiers Context	31
1.16.8	Object Data Context	31
1.16.9	Material Context	31
1.16.10	Texture Context	31
1.16.11	Particles Context	31
1.16.12	Physics Context	31
1.16.13	Where Did The Old Stuff Go?	31
1.17	3D View Windows	31
1.17.1	The Viewport and its Contents	31
1.17.2	Modes	33
1.17.3	Viewport Options	34
1.17.4	Changing Your Viewpoint, Part One	35
1.17.5	Positioning the 3D Cursor	36
1.17.6	Changing Your Viewpoint, Part Two	37
1.17.7	View Navigation	38
1.17.8	Visibility Layers	39
1.17.9	Count Your Polys	40
1.18	Object Mode	40
1.18.1	Introduction	40
1.18.2	Object Origin	40
1.18.3	Multiple Selections	40
1.18.4	Selecting Obscured Objects	41
1.18.5	Selecting Everything and Nothing	41
1.18.6	Hiding Things	41
1.18.7	Local Versus Global View	41
1.18.8	Border Select (Box Selection)	41
1.18.9	Circle Select (Brush Selection)	41
1.18.10	The Manipulator	41
1.18.11	Transformation Hotkeys	43
1.18.12	Choosing the Pivot Point	43
1.18.13	Basic Camera Technique	44
1.18.14	Adding/Removing Objects, Undo/Redo, Repeat	45
1.18.15	Assigning Layers	45
1.18.16	Object, Action, Settings	45
2	Unit 2 - Basic Modeling and Shading	46
2.1	Meshes and Edit Mode	46
2.1.1	What Is a Mesh?	46
2.1.2	Introduction to Edit Mode	46

2.1.3	Selection Modes	47
2.1.4	Multiple Selections	47
2.1.5	Hiding Things	48
2.1.6	Local Versus Global View	48
2.1.7	Border Select (Box Selection) & Circle Select (Brush Selection)	48
2.1.8	Select More, Select Less	48
2.1.9	Manipulator, Transformation Hotkeys, Pivot Point	48
2.1.10	Proportional Editing	49
2.1.11	Deleting Things	49
2.1.12	Undo/Redo	50
2.1.13	Adding Things	50
2.1.14	Simplifying Things	51
2.2	Normals and Shading	52
2.2.1	Not So Smooth?	53
2.3	More Mesh Editing Techniques	53
2.3.1	Adding More Mesh Pieces	54
2.3.2	Linked Selections	54
2.3.3	Separating and Joining Meshes	54
2.3.4	Proper Extrusion	54
2.3.5	Edge Loop Selection	54
2.3.6	Loop Cuts	55
2.3.7	Edge Loop Deletion	55
2.3.8	Subdividing Parts	55
2.3.9	Subdivision Surface Modifier	55
2.3.10	Sharpening the Curves	56
2.4	Quickie Lighting	58
2.5	Quickie Model	59
2.5.1	Bring up the Default Cube	59
2.5.2	Setting up the Viewport	59
2.5.3	Adjusting the Height	60
2.5.4	Extruding	61
2.5.5	Merging	61
2.5.6	Saving your Work	62
2.5.7	Additional Resources	62
2.6	Quickie Render	62
2.6.1	Rendering the Quickie Model	62
2.6.2	Seeing Your Render	63
2.6.3	Aiming the Camera	63
2.6.4	Lighting	63
2.6.5	Saving the Render	63
2.6.6	Renderer Selection	63

2.6.7	Render Control	64
2.6.8	Render Image Dimensions	64
2.6.9	Image File Formats	65
2.6.10	Additional Resources	65
2.7	Enter the World	65
2.8	Understanding the Camera	66
2.8.1	Real-World Cameras	66
2.8.2	The Camera In Blender	67
2.8.3	See Also	68
2.9	Improving Your House	69
2.9.1	Adding a Ground Plane	69
2.9.2	Scaling with a Pivot	70
2.9.3	Edge Selection	70
2.9.4	Extruding Edges	70
2.9.5	Face Selection	71
2.9.6	Extruding Faces	72
2.9.7	Deleting Edges	72
2.9.8	Subdividing Faces	73
2.9.9	Final Steps	74
2.10	Extruding a Simple Person	74
2.10.1	Start a New Scene	74
2.10.2	Selection Methods	74
2.10.3	Extruding Limbs	75
2.10.4	Adding the Head	77
2.10.5	Save Your Work	78
2.11	Smoothing Your Simple Person	78
2.11.1	Subsurfaces	79
2.11.2	Smooth Shading	80
2.12	Improving Your Simple Person	81
2.12.1	Widening the Torso	81
2.12.2	Bending the Arms	81
2.12.3	Making Feet	82
2.12.4	Reshaping the Head	83
2.13	Spinning a Simple Hat	83
2.13.1	Creating a Generatrix	83
2.13.2	Spinning the Hat	84
2.13.3	Smoothing Your Hat	84
2.13.4	Additional Resources	85
2.14	Putting the Hat on the Person	85
2.14.1	Adjusting an Object's Median Point	85
2.14.2	Positioning the Hat	86

2.14.3 Parenting the Hat to the Person	86
2.14.4 Renaming Objects	87
2.14.5 Outliner Windows	87
2.14.6 Good on ya' mate!	87
2.14.7 Additional Resources	87
2.15 Overview	88
2.15.1 Material versus Texture	88
2.15.2 Other Material Settings	88
2.15.3 Types of Textures	88
2.15.4 Additional Resources	88
2.16 Quickie Material	89
2.16.1 Your First Material	89
2.16.2 Specifying Colors	90
2.16.3 Additional Resources	91
2.17 Multiple Materials per Object	91
2.17.1 Set the Scene	92
2.17.2 Colorize Time	92
2.18 Metal Versus Plastic	93
2.18.1 Making It Plastic	93
2.18.2 Making It Metal	93
2.18.3 Making It Ceramic	94
2.19 Texture Settings	94
2.19.1 Material Textures	95
2.19.2 World Textures	95
2.20 Image Textures	96
2.20.1 Image Texture Settings	96
2.20.2 Making Your Own Texture	97
2.21 Procedural Textures	101
2.21.1 A Simple Wood Texture	101
2.22 Quickie Texture	102
2.22.1 Making It Mottled	102
2.22.2 Making It Bumpy	103
2.22.3 Some Closing words	103
2.23 Halo Materials	104
2.23.1 Introduction	104
2.23.2 Setting The Scene	104
2.23.3 Adding The Flare	104
2.23.4 The Final Result	105
2.24 Blender Memory Management	105
2.24.1 Datablocks And Users	105
2.25 Using Bones	106

2.25.1 Laying down bones	106
2.25.2 Add a bone	106
2.25.3 Extrude a second Bone	107
2.25.4 Name the bones	107
2.25.5 Parent the bones	107
2.25.6 Moving the Bones	107
2.25.7 In-Depth Info on Selected Bone Topics	108
2.26 Mountains out of Molehills	108
2.26.1 Creating a simple plane	108
2.26.2 First mountain	109
2.26.3 Peaks vs. hills	109
2.26.4 Shaping the world	110
2.26.5 Smoothing things out	111
2.26.6 Naturalness	111
2.27 Modeling a Volcano	112
2.27.1 Adding a Plane	112
2.27.2 Making the Mountain	112
2.27.3 Forming the Crater	114
2.27.4 Finishing the crater	114
2.27.5 Adding Magma	115
2.27.6 Varying the Terrain	116
2.28 Penguins from Spheres	117
2.28.1 Setup	117
2.28.2 Creating the body	117
2.28.3 Shaping the head	118
2.28.4 Extruding the wings	120
2.28.5 Smoothing the wings	121
2.28.6 Cutting the underside	122
2.28.7 Adding the feet	123
2.28.8 Extruding a tail	124
2.28.9 Subsurfing	124
2.28.10 Extra	125
2.29 Dicing With Depth (Dice Modeling)	125
2.29.1 The Basic Mesh	126
2.29.2 Making the Pips	127
2.29.3 Rounding Things Off	128
2.29.4 Colouring Your Die	129
2.30 Model a Goblet	129
2.30.1 Three ways to build the mesh	129
2.30.2 How many circles?	130
2.31 Model a Silver Goblet	130

2.31.1 Basic Shape	130
2.31.2 Smoothing and Defining	131
2.31.3 Quick links for the impatient	133
2.32 Model a Silver Goblet Another Way	133
2.32.1 Preliminaries	133
2.32.2 Description of the modeling steps	133
2.32.3 Creating the Goblet	133
2.32.4 Subsurfing and smoothing the goblet	134
2.32.5 Flattening the base of the goblet	134
2.33 Spin a goblet	135
2.33.1 Modeling	135
2.33.2 Rendering	136
2.34 Light a Silver Goblet (Early look at lighting)	137
2.34.1 Techniques	137
2.34.2 Objects in the Scene	137
2.34.3 Adding the Atmosphere	138
2.34.4 Creating a metallic texture for the goblet:	139
2.35 Simple Vehicle	140
2.36 Simple Vehicle: Wheel	141
2.36.1 Techniques	141
2.36.2 Model the tire	141
2.36.3 Model the hubcap	142
2.36.4 Extra	144
2.37 Simple Vehicle: MudTires	144
2.37.1 Build the Tire	144
2.37.2 Finish your tire several versions	145
2.37.3 Finishing	147
2.37.4 Colouring	147
2.37.5 Creating a more complex tire	148
2.38 Simple Vehicle: Seat	151
2.38.1 Techniques	151
2.38.2 Extrude the Seat	151
2.38.3 Add cushion seams	151
2.38.4 Position the cushion seams	152
2.38.5 Add Depth to the seams	152
2.38.6 Subsurf the seat	152
2.38.7 Resize the seat	153
2.38.8 Final touches	154
2.39 Simple Vehicle: Rocket Launcher	155
2.39.1 Techniques	155
2.39.2 Overview	155

2.39.3 Create the Launcher	155
2.39.4 Create the rocket	155
2.39.5 Create the mount	156
2.39.6 Subsurf	156
2.39.7 Final touches	157
2.40 Simple Vehicle: Body	157
2.40.1 Techniques	157
2.40.2 Planning	157
2.40.3 Building the Jeep	157
2.40.4 A Touch of Detail	164
2.40.5 Subsurf	168
2.40.6 Optional Activities	168
2.41 Simple Vehicle: Some Assembly Required	168
2.41.1 Techniques	168
2.41.2 Overview	168
2.41.3 Appending the File	168
2.41.4 Rinse and Repeat	169
2.41.5 Parenting	169
2.41.6 Final things	169
2.41.7 Extra	170
2.41.8 Additional Tutorials	171
2.42 Modeling a 3D Parachute in Blender	171
2.42.1 Remove the default cube	171
2.42.2 Add a cylinder	171
2.42.3 Remove the bottom row of vertices	172
2.42.4 Extrude and scale to shape	172
2.42.5 Make the top more rounded	173
2.42.6 Extrude the parachute straps	173
2.42.7 Merge it together	173
2.43 Model a Low Poly Head	174
2.43.1 Overview	174
2.43.2 Add a plane	174
2.43.3 Make a pointed chin	174
2.43.4 Extrude the face	175
2.43.5 Make the facial features	176
2.43.6 Smooth the mouth region	178
2.43.7 Make the head	179
2.43.8 Make the back of the head	179
2.43.9 Finishing it up	180
2.44 Building a House	182
2.44.1 Setting the Scene	182

2.44.2 Make the Roof	182
2.44.3 Naming the Roof and House Objects	184
2.44.4 Colouring the House	184
2.44.5 Make a Chimney	184
2.44.6 Adding a Window	185
2.44.7 Adding a Door	187
2.44.8 Building the Fence	188
2.44.9 The Ground Plane and a Path	192
2.44.10 Improve the Lighting	193
2.44.11 Check the Camera View	193
2.44.12 Final Render	193
2.45 Pipe joints	194
2.45.1 T joint	194
2.45.2 6 cylinder joint	194
2.45.3 3 cylinder joint	195
2.45.4 T joint with smaller diameter	195
2.46 Lighting Suzanne: Introductory one lamp lighting	199
2.47 Lighting Suzanne: Introductory one lamp lighting	199
2.47.1 Suzanne, Our Star	199
2.47.2 The default light	199
2.47.3 Types of lamps	199
2.47.4 Light Energy	200
2.47.5 Distance and Height	200
2.47.6 Rotation	200
2.47.7 Experiment!	201
2.47.8 Note on Transparency	201
2.47.9 Note on World Lighting	201
2.48 Overview	202
2.48.1 Why Use Curves?	202
2.48.2 Bézier Versus NURBS	202
2.48.3 More Than You Wanted To Know About Curves	202
2.49 Intro to Bézier Curves	202
2.49.1 Bézier Curves	202
2.49.2 NURBS Curves	204
2.49.3 Which Curves To Use?	205
2.49.4 Extruding from 2D to 3D	205
2.49.5 Make a Simple Face of Bézier Curves	205
2.50 Bevelling a Curve	207
2.50.1 Built-In Bevel	207
2.50.2 Custom Bevel	209
2.50.3 Custom Taper	209

2.51 NURBS Patches	210
2.51.1 Your First NURBS Patch	210
2.52 Deforming Meshes using the Curve Modifier	211
2.52.1 Understanding the Curve Modifier	211
2.52.2 Another Example	213
2.53 The Empty Object	215
2.53.1 Using an Empty With the Array Modifier	215
2.54 Background Images	217
2.54.1 Other Ways To Bring In Reference Images	218
2.55 Aligning Vertices with a Guide Image	218
2.55.1 Background: orthographic projection	218
2.55.2 Making a Simple Pyramid	219
2.55.3 Using the guide images	219
2.56 Modeling a Fox from Guide Images	222
2.56.1 Method 1	222
2.56.2 Method 2 (using mirroring)	227
2.56.3 Using mirroring with method 1	228
2.56.4 Detailed steps to align images using GIMP	228
2.57 Using Bézier Curve to Model a 3D logo from a 2D logo	229
2.57.1 Overview	229
2.57.2 Load the background image into blender	229
2.57.3 Delete the cube	230
2.57.4 Introducing the Bezier Curve	230
2.57.5 Modeling the lighting bolt	231
2.57.6 Modeling the red circle	235
2.58 Subsurface Scattering	243
2.59 Ray Tracing	244
2.59.1 Setting The Scene	244
2.59.2 Ray-Traced Transparency	245
2.59.3 The Fresnel Factor	246
2.59.4 Ray-Traced Mirroring	246
2.59.5 Why Are My Shadows Black?	246
2.60 Using Textures	249
2.60.1 Colormap	249
2.60.2 Diffusemap	249
2.60.3 Luminositymap	250
2.60.4 Specularitymap	250
2.60.5 Reflectionmap	251
2.60.6 Transparencymap	251
2.60.7 Refractionmap	251
2.60.8 Translucencymap	251

2.60.9 Bumpmap/Normalmap	251
2.60.10 Displacement Map	252
2.60.11 Dirtmap	252
2.60.12 Use UV Coordinates for your Maps	252
2.61 Using a texture to make a material partially transparent	253
2.61.1 Using a greyscale texture - white for opaque	253
2.61.2 Using a partially transparent texture to make the material transparent	254
2.62 Creating Basic Seawater	255
2.62.1 Create 3 linked planes	255
2.62.2 Create material	256
2.62.3 Create Material / Textures (Blender 2.63)	256
2.62.4 Extra Practice	257
2.63 Mountains Out Of Molehills 2	257
2.64 Basic Carpet Texture	259
2.64.1 Goal	259
2.64.2 The Basic Material and Shader Settings	259
2.64.3 Cloud Texture 1	260
2.64.4 Cloud Texture 2	260
2.64.5 The Final Cloud Texture	261
2.64.6 The Last Texture	261
2.65 The Rusty Ball	262
2.66 Creating Pixar-looking eyes	263
2.66.1 Parts of the Eye	263
2.66.2 Materials	264
2.66.3 Lighting	265
2.66.4 Changing the Eye Color	266
2.66.5 Pixar Eye Video Tutorial	266
2.66.6 Useful Links	266
2.67 Procedural Eyeball	267
2.67.1 Creating the Mesh	267
2.67.2 Adding the Textures	269
2.67.3 Pupil Dilation	274
2.68 Putting It All Together: A Dragon!	276
2.69 Putting it all Together: A Dragon!	276
2.69.1 Modeling from the Image	276
2.69.2 Setting the Body Materials and Textures	277
2.69.3 Adding the Eyes	277
2.69.4 A Dragon Needs Teeth and Claws	277
2.69.5 Horns, a Crest, and Nostrils	278
2.69.6 Giving the Dragon Wings	279
2.69.7 FIRE!	279

2.69.8	Ground to Stand on	280
2.69.9	Lighting	280
2.70	THE FINAL RESULT	281
2.71	Noob Project No.02 - Space Dragon in Space	281
3	Unit 3 - Broadening Horizons	284
3.1	The UV/Image Editor	284
3.1.1	Viewing Render Results	284
3.1.2	Viewing/Editing Texture Images	284
3.1.3	UV Mapping	284
3.2	UV Map Basics	284
3.2.1	The Basics of UV Mapping	285
3.2.2	Alternative method	287
3.2.3	Learning More	288
3.3	Realistic Eyes In Blender	291
3.3.1	Overview	291
3.3.2	Reference Material	291
3.3.3	Building The Eye	291
3.3.4	UV Mapping	292
3.3.5	Preparing A Template For Texturing	295
3.3.6	Bump Map - The Sclera	296
3.4	Beginning Lighting	297
3.5	Understanding Real Lights	297
3.5.1	The Properties of Real Lights	297
3.5.2	298
3.6	Understanding Blender Lights	298
3.6.1	Light Settings	298
3.7	Basic Lighting Rigs	299
3.7.1	One-Point Lighting	299
3.7.2	Two-Point Lighting	300
3.7.3	Three-Point Lighting	300
3.7.4	Other Lighting Setups	301
3.7.5	See Also	301
3.8	Faked Global Illumination with Blender internal	301
3.8.1	Preamble	301
3.8.2	Blender Faked GI Tutorial	301
3.9	Practising Good Parenting	305
3.9.1	Parenting and Unparenting	305
3.9.2	Lock Up Your Children!	306
3.9.3	Example: Camera Pan	306
3.10	Overview	306
3.10.1	The Timeline	307

3.10.2 See Also	307
3.11 Introduction to Keyframing	307
3.11.1 First Keyframes	307
3.11.2 Preview From All Angles	307
3.11.3 What Is Being Animated?	308
3.11.4 Animating a Material Property	308
3.12 The Ways of the Animator	308
3.13 Animation Editors	309
3.13.1 Timeline	309
3.13.2 Graph Editor	309
3.13.3 Dope Sheet	309
3.13.4 NLA Editor	309
3.13.5 To Summarize...	309
3.14 Introducing the Graph Editor	310
3.15 Animation Rendering	311
3.15.1 Render Control	311
3.15.2 Image Dimensions	311
3.15.3 Image File Formats	312
3.15.4 Single Frame Versus Entire Animation	313
3.16 Lattice Modifier	313
3.16.1 What is a Lattice?	313
3.16.2 How to add and use a Lattice	313
3.16.3 Getting more involved with Lattice	314
3.16.4 Examples	316
3.16.5 Making it stick	316
3.16.6 Animating a Lattice	317
3.16.7 Let the fun begin!	321
3.17 Bouncing Ball with Lattice	321
3.17.1 How to make a ball bounce convincingly!	321
3.17.2 How hard can it be to move a ball?	321
3.17.3 Why use a lattice?	321
3.17.4 Let's begin then	321
3.17.5 Saving The Animation	324
3.17.6 So what next?	324
3.17.7 Links	325
3.18 Creating Basic Water Animation	325
3.18.1 Water and Other Fluids	325
3.18.2 The Domain	325
3.18.3 The Fluid	325
3.18.4 The Obstacle	326
3.18.5 Baking Fluids	326

3.18.6 Finishing touch	326
3.18.7 Other Fluid Objects	326
3.19 Flying Through a Canyon	327
3.19.1 Creating the Canyon	327
3.19.2 Coloring the Canyon	330
3.19.3 Guide the camera through the canyon	331
3.19.4 Additional projects	332
3.20 Using the Sequencer to Compile Frames into an Animation	332
3.20.1 Preamble	332
3.20.2 Tutorial	332
3.20.3 Use the Sequencer to compile the images into a movie file	332
3.21 Further Rendering Options	333
3.21.1 Stamping	333
3.21.2 Toon Renders	333
3.21.3 Clay Renders	334
3.21.4 Transparent Backgrounds	334
3.22 Overview	334
3.22.1 Introduction	334
3.22.2 The very first particle system	334
3.23 Fire	337
3.23.1 The particle system	337
3.23.2 Material	338
3.23.3 Rendering	339
3.24 Fur	339
3.24.1 The emitter	339
3.24.2 Material	341
3.24.3 Changing Hair length with a texture	342
3.24.4 Comb it!	344
3.24.5 Links	344
3.25 Fireworks	344
3.25.1 The Emitter	344
3.25.2 Reactor 1	345
3.25.3 Smoke Trail	346
3.25.4 Render	346
3.26 Particles forming Shapes	346
3.26.1 Keyed Physics	347
3.26.2 Harmonic Force Fields	350
3.27 Billboard Animation	351
3.27.1 Splitting a texture	351
3.27.2 Animating Alpha in relative particle time	352
3.27.3 Animating billboard color	352

3.27.4	Changing the starting color of a billboard	353
3.28	Soft Body Animation	353
3.28.1	Explanation of Settings	355
3.29	Simple Cloth Animation	355
3.29.1	Making the Skirt Mesh	355
3.29.2	Creating the Vertex Group	355
3.29.3	Animating the Skirt	355
3.29.4	Prior to Keeping the Folds	356
3.29.5	Keeping the Folds	356
3.29.6	Extra Practice	357
3.30	Soft Body with Wind	357
3.30.1	Prologue	357
3.30.2	Setting up scene	357
3.30.3	Designating each object's job	358
3.30.4	The movement	359
3.30.5	Finishing touches	361
3.30.6	Final result	362
3.31	Blender Game Engine Basics- Rolling Ball	363
3.31.1	Introduction	363
3.31.2	Adding the Hill	363
3.31.3	Creating the Ball	363
3.31.4	Testing your game	363
3.31.5	Video capturing your game	364
3.31.6	Conclusion	364
3.31.7	Extra Tutorials	364
3.32	Platformer: Controls and Movement	364
3.32.1	Set Up	364
3.32.2	Controls	365
3.33	Maze: Force and Multiple Levels	365
3.33.1	Introduction	365
3.33.2	Maze Surface	366
3.33.3	Character	366
3.33.4	Motion	367
3.33.5	Testing	368
3.33.6	Falling	368
3.33.7	Camera	369
3.33.8	Beautification	370
3.33.9	Levels	371
3.33.10	Dynamic obstacles	372
3.33.11	Conclusion	372
3.34	Platformer: Improving the Physics	372

3.34.1	Introduction	372
3.34.2	Creating Your Hit Test Object	372
3.34.3	Excessive Bouncing	373
3.34.4	Final Notes	373
3.35	How to Make an Executable	373
3.35.1	Windows	373
3.35.2	GNU/Linux or Mac OS X	374
3.35.3	BlenderPlayer	374
3.35.4	Legal Issues	375
3.35.5	Proprietary Blend Files	375
3.36	Build a Skybox	375
3.36.1	Prologue	375
3.36.2	Gather your graphics	375
3.36.3	Create a dome for the sky	376
3.36.4	Create a dome for the ground	377
3.36.5	Render the environment map	378
3.36.6	Video Tutorial	379
3.37	Basic Mouse Pointer	380
3.37.1	Mouse Pointer	380
3.38	Text in BGE	380
3.38.1	Text	380
3.38.2	Links	380
3.39	Platformer: Creating the Engine with Python	380
3.39.1	Introduction	380
3.39.2	Code Sections	381
3.39.3	Adding the Player Object	381
4	Unit 4 - Taking Off with Advanced Tutorials	383
4.1	Introduction	383
4.1.1	First Steps In Blender Scripting	383
4.1.2	The bpy Module	384
4.1.3	The mathutils Module	384
4.1.4	See Also	384
4.2	Anatomy Of An Addon	384
4.2.1	Introduction	384
4.2.2	Text Blocks	384
4.2.3	Your First Operator	384
4.2.4	Invoking Your Operator	386
4.2.5	If You Hit An Error	386
4.3	A User Interface For Your Addon	386
4.3.1	Start	386
4.3.2	Your First Panel	386

4.3.3	Adding Undo Support	387
4.3.4	Put It All Together	387
4.3.5	Space Types, Region Types, Contexts, Oh My!	387
4.4	Adding A Custom Property	387
4.4.1	Defining A Property	387
4.4.2	Using The Property	388
4.4.3	Put It All Together	388
4.5	A Separately Installable Addon	389
4.5.1	The Addon Info Dictionary	389
4.5.2	Making The Addon Installable	389
4.5.3	Installing The Addon	389
4.5.4	Using The Installed Addon	389
4.6	Object, Action, Settings	390
4.6.1	Prologue	390
4.6.2	Adding To The Add Menu	390
4.6.3	Fixing Up The UI	390
4.6.4	invoke Versus execute	391
4.6.5	Put It All Together	391
4.6.6	Undocumented Blender	391
4.7	Overview	392
4.7.1	Advanced Modeling	392
4.8	High Dynamic Range imaging (HDRi)	392
4.8.1	Introduction	392
4.8.2	Definitions	392
4.8.3	Usage	392
4.8.4	Quick Tutorial (<i>for experienced blendies</i>)	392
4.8.5	Step-by-step Tutorial	393
4.9	Creating a Light Probe	394
4.10	Landscape Modeling with Heightmaps	395
4.10.1	Creating the heightmap image	395
4.10.2	Create grid and add the image as texture	396
4.10.3	Use texture as heightmap	396
4.11	How to Do Procedural Landscape Modeling	398
4.11.1	Mountains	398
4.12	Landscape Modeling I: Basic Terrain	401
4.12.1	Introduction	401
4.12.2	Creating the Canvas	401
4.12.3	Molding the Mountains	401
4.12.4	Texturing the Terrain	403
4.12.5	Reader Contributions	404
4.12.6	Reader Contributions 2	405

4.12.7 Reader Contributions 3	405
4.13 Landscape Modeling II: Texture Stenciling	405
4.13.1 UPDATED FOR 2.7 USERS	405
4.13.2 Creating the Stencil	406
4.13.3 Applying the Stencil	407
4.13.4 Adding Snow	408
4.13.5 Multiple Stencils	409
4.13.6 For 2.7 blender users:	410
4.14 Landscape Modeling III: Exporting as a Heightmap	411
4.14.1 A Word About Heightmaps	411
4.14.2 Creating the Material	411
4.14.3 Setting up the Camera	413
4.14.4 Avoid Stair-Stepping	414
4.15 Bump Mapping	414
4.15.1 Creating Bump Maps	414
4.15.2 Applying the animated bump map	416
4.16 Normal Mapping	416
4.17 Texture Normals	417
4.17.1 The Texture-Normal	417
4.18 Color Map Normals	419
4.18.1 The Color-Map-Normal	419
4.18.2 Creating a Normal Map	419
4.19 Introduction	421
4.20 Texture Nodes	421
4.20.1 A Simple “Rainbow” Texture	421
4.20.2 Scalars Versus Vectors	422
4.21 Material Nodes	423
4.21.1 A Simple Graduated Material	423
4.22 Compositing	425
4.22.1 Compositing Example	425
4.22.2 Things to Try	427
4.22.3 Other Tutorials	427
4.23 Further Compositing: A Portal Effect	427
4.23.1 The Outer Scene	427
4.23.2 Animating The Camera	428
4.23.3 The Inner Scene	428
4.23.4 Inner Scene Animation	428
4.23.5 Compositing It All Together	429
4.23.6 More On Lighting	429
4.24 Advanced Rendering	429
4.25 Alternative Renderers	429

4.25.1 Cycles	429
4.25.2 Freestyle	430
4.25.3 External Renderers	430
4.26 Disadvantages of Other Renderers	430
4.26.1 Say Goodbye to Your Materials and Lighting	430
4.26.2 High-Quality Renderers Are S-L-O-W	430
4.27 Introduction to Cycles	430
4.27.1 A Simple Cycles Example	430
4.27.2 Real-Time Rendering Previews	432
4.27.3 Nodes? What Nodes?	433
4.27.4 See Also	433
4.28 A Glass Material in Cycles	434
4.28.1 The Magic Sky	434
4.28.2 Here Comes The Glass	436
4.28.3 Adjustable Refraction	436
4.28.4 A Reusable Custom Shader	437
4.29 Dealing with Fireflies in Cycles	439
4.29.1 Making An Example	439
4.29.2 Fixing The Problem 1: Renderer Tweaks	441
4.29.3 Fixing The Problem 2: The Despeckle Filter	441
4.30 Fireflies in Cycles, Continued	442
4.30.1 Finer Despeckling: Separate Lighting Passes	443
4.30.2 Conclusion	446
4.30.3 See Also	446
4.31 Procedural Eyeball in Cycles	446
4.31.1 Creating the Mesh	446
4.31.2 UV-Mapping the Iris	447
4.31.3 Creating the Materials	448
4.31.4 The Final Result	449
4.32 Introduction to Freestyle	450
4.32.1 A Simple Freestyle Example	450
4.32.2 See Also	451
4.33 Overview	451
4.33.1 Index	451
4.34 Introduction	451
4.35 Guided Tour:	452
4.36 Armature Object	452
4.37 Armature Object in Object Mode	453
4.37.1 The Armature Object	453
4.37.2 The Edit Panel When in Object Mode	453
4.37.3 Extra Practice	455

4.38 Armature Object in Edit Mode	455
4.38.1 Now the basics about bones	455
4.38.2 The edit panel	456
4.38.3 Naming convention	458
4.38.4 Mirror Editing	459
4.39 Armature Object in Pose mode	461
4.39.1 So What Can You Do?	461
4.40 Mesh Object	462
4.41 Connection between Armature and Mesh	462
4.41.1 The Armature Modifier	462
4.41.2 The Old Way	463
4.41.3 Tip: Bake envelope to vertex groups	463
4.42 Envelope	463
4.42.1 What is Envelope	463
4.42.2 Edit Envelope	463
4.42.3 Envelope Options	464
4.43 Vertex Groups & Weight Paint	465
4.43.1 What Are Vertex Groups?	465
4.43.2 Weight Paint	465
4.43.3 Vertex Groups and Armatures	466
4.43.4 Using Weight Painting with Armatures	467
4.44 Shape Keys	468
4.44.1 Shape Key?	468
4.44.2 The GUI	469
4.44.3 Shape Keys step-by-step	469
4.45 Lip-Sync with Shape Keys	469
4.45.1 Lip-Sync with Shape Keys	469
4.45.2 The hard work first	470
4.45.3 Getting down and dirty	472
4.45.4 Putting it all together	473
4.45.5 Using Blender to render Audio AND Video TOGETHER.	474
4.45.6 Audio Files	474
4.45.7 Note	474
4.46 Constraints	474
4.46.1 The Constraint	474
4.46.2 The Constraint Panel	474
4.46.3 The Constraint Index	475
4.47 Copy Location	475
4.47.1 Copy Location	475
4.47.2 The Constraint Panel	475
4.47.3 Where To Use It	475

4.48 Copy Rotation	476
4.48.1 Copy Rotation	476
4.48.2 The Constraint Panel	476
4.48.3 Where To Use It	476
4.49 Track-To	476
4.49.1 Track-To	476
4.49.2 The Constraint Panel	476
4.49.3 Where To Use It	477
4.50 Floor	477
4.51 Locked Track	477
4.51.1 Locked Track	477
4.52 Follow Path	477
4.52.1 Follow path	477
4.52.2 The Constraint Panel	477
4.52.3 Where To Use It	478
4.53 Stretch-To	478
4.53.1 Stretch-To	478
4.54 IK Solver	478
4.54.1 The IK solver	478
4.54.2 The Constraint Panel	479
4.54.3 Where To Use It	480
4.54.4 Degree Of Freedom	480
4.55 Timeline Window	480
4.55.1 Timeline Window	480
4.56 IPO Window	480
4.56.1 Graph Editor	480
4.57 Data Type	480
4.57.1 Dope Sheet	480
4.58 NLA Window	480
4.59 The NLA Window	480
4.59.1 Walkthrough	481
4.60 Introduction To NLA Editor	481
4.60.1 NLA (Non-Linear Animation)	481
4.60.2 Setting up the Scene	482
4.60.3 Adding Action Strips	483
4.60.4 Getting the Strips to Play	483
4.60.5 Conclusion	484
4.61 The Stride feature	484
4.62 Relative Vertex Keys	484
4.62.1 Introduction:	484
4.62.2 Window Layout:	484

4.62.3 Setting your Neutral Pose	484
4.62.4 Setting up your additional Pose Lines	484
4.62.5 Set your Poses	484
4.62.6 Name your Poses	484
4.62.7 Time to Animate (b)	484
4.62.8 Adjust your Slow in & Out	485
4.63 Working Example: Bob	485
4.63.1 Getting Started	485
4.64 Building the Rig	485
4.65 Deform the Mesh	486
4.66 Create a Walk Cycle	486
4.67 Working example: Piston, Rod and Crank	486
4.67.1 Piston, Rod and Flywheel	486
4.67.2 Links	489
4.68 Working example: Cutting Through Steel	489
4.68.1 Creating the Alpha-Map	489
4.68.2 Welding sparks	491
4.68.3 Using the alpha map on an object	491
4.68.4 Material for the plane	492
4.68.5 Lighting the torch	493
4.68.6 More realistic sparks	494
4.68.7 Smoke	494
4.68.8 After Glow	495
4.69 Overview	496
4.70 Advanced Game Engine Techniques (GUI)	496
4.70.1 Making Blender Games Graphically	496
4.71 Creating Pop-Up Menus	497
4.71.1 Making An Easy Title Menu	497
4.71.2 Author's Starting Menu Example	498
4.71.3 A Main Menu	498
4.72 Creating Dropping Menus	500
4.72.1 Making Moving Selectors	500
4.72.2 Make the Menu Move	502
4.73 The "5-Layer" Button	502
4.73.1 The 5-Layer Button	502
4.74 Creating Object Outlines	502
4.74.1 Creating Object Outlines	502
4.75 Advanced Game Engine Techniques (Python)	502
4.75.1 Game Creation Techniques (Python)	502
4.75.2 Additional Resources	502
4.75.3 External Links	502

4.76 Hacking Blender	502
4.76.1 Getting the Blender Source Code	502
4.76.2 Layout of the Blender Source	503
4.76.3 Blender’s “Genetic Code”: “DNA” and “RNA”	504
4.76.4 Special Globals: “G” and “U”	504
4.76.5 Naming Conventions	504
4.76.6 User-Interface Implementation	504
4.76.7 See Also	504
4.77 Introduction to Game Engine Source	504
5 Appendices	506
5.1 Glossary	506
5.1.1 A	506
5.1.2 B	506
5.1.3 C	507
5.1.4 D	507
5.1.5 E	507
5.1.6 F	507
5.1.7 G	508
5.1.8 H	508
5.1.9 I	508
5.1.10 J	508
5.1.11 K	508
5.1.12 L	508
5.1.13 M	509
5.1.14 N	509
5.1.15 O	509
5.1.16 P	509
5.1.17 Q	509
5.1.18 R	510
5.1.19 S	510
5.1.20 T	511
5.1.21 U	511
5.1.22 V	511
5.1.23 W	511
5.1.24 X	511
5.1.25 Y	511
5.1.26 Z	511
5.2 Frequently Asked Questions	511
5.3 Tutorial Links List	512
5.3.1 General Blender Information	512
5.3.2 Blender Tutorial Websites	512

5.3.3	Blender Interface	513
5.3.4	Modeling	513
5.3.5	Texturing	514
5.3.6	Lighting, Shadows and Rendering	514
5.3.7	Compositing	514
5.3.8	Animation	514
5.3.9	Particles	515
5.3.10	Fluid Simulation	515
5.3.11	Game Engine	515
5.3.12	Python and Plugins	515
5.3.13	Related Programs	516
5.3.14	Blender WikiBooks	516
5.3.15	FAQ	516
5.3.16	Miscellaneous	516
5.3.17	Other Lists	516
5.3.18	About	516
5.4	Hotkeys	516
5.5	Output Formats	517
5.6	Image Portfolio	517
5.6.1	Summary	517
5.6.2	Categories	518
5.6.3	A sampling of images from the above categories	518
5.7	Blender Glossary	524
5.8	Materials Directory: Every Material Known To Man	526
5.8.1	“Every Single Material Known To Man” Project	526
5.8.2	Metals and Minerals	527
5.8.3	Cloth and Fabrics	527
5.8.4	Particle-Based	527
5.8.5	Earth and Home	527
5.8.6	SSS-Used	528
5.8.7	Other	528
5.8.8	Abstract	528
5.8.9	Human	528
5.8.10	Elemental	528
5.8.11	External material libraries	528
5.9	Sources of free 3D models	528
5.9.1	Blender-Specific	528
5.9.2	Non-Blender-Specific	528
5.10	All Blueprints Links	530
5.10.1	Registration Required	530
5.10.2	Payment Required	530

5.10.3 Shouldn't Really Be Here?	530
5.11 Materials, Textures, Photos	530
5.12 Asking for Help	531
5.13 Tips for a Successful Project	531
5.14 Modeling Realistically	532
5.14.1 Assemble Things, Don't Chisel Them	532
5.14.2 Separate Objects for each Material	532
5.14.3 Removing the Fourth Wall	533
5.14.4 Proportions and Measurements	533
5.15 Modeling tips	533
5.16 Cheat the 3D	534
5.16.1 Ways to Improve Performance	534
5.17 Performance vs. Quality	534
5.18 Modeling a Gingerbread Man	535
5.18.1 Modeling	535
5.18.2 Camera Positioning and Rendering	540
5.18.3 Applying Textures	541
5.19 Modeling a simple space-ship	543
5.19.1 Modeling a simple spaceship using Blender 2.49b	543
5.20 GIMP)	543
5.21 Part 1 - Preparing the Scene	543
5.21.1 Start a new scene in Blender	543
5.21.2 Prepare the individual particles	543
5.21.3 Defining the motion of the particles	544
5.22 Creating Weapons based on 2D Images	545
5.22.1 Modeling Technique 1	545
5.22.2 Modeling Technique 2	546
5.23 Modeling with Meta Balls	546
5.23.1 Getting started	546
5.23.2 Sculpting With “Lumps of Clay”	546
5.23.3 Meta-mess!	547
5.23.4 Beginning to sculpt	547
5.24 Match Moving	550
5.24.1 Icarus	550
5.24.2 Voodoo	550
5.25 Match Moving/Motion Tracking with Icarus and Blender	550
5.26 Create a Clayman	551
5.26.1 Create a Clayman	551
5.26.2 Making The Clayman	551
5.26.3 Adding Materials	552
5.26.4 Inserting Armatures	552

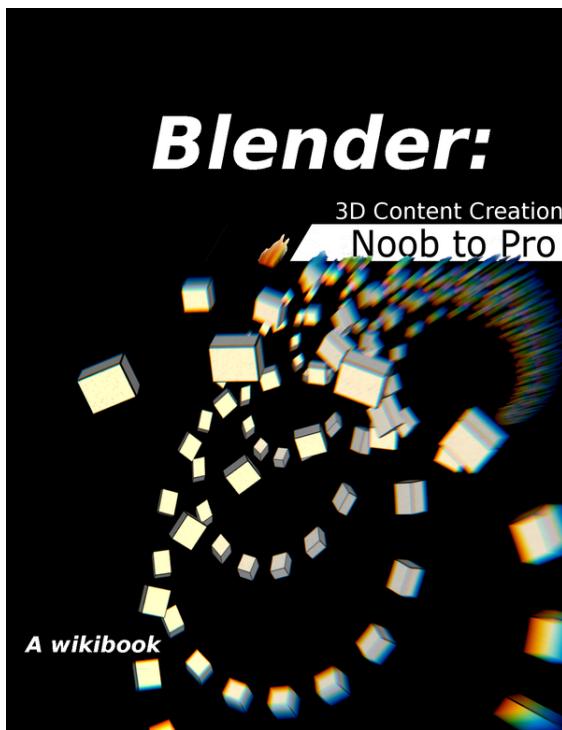
5.26.5 Applying The Armature	552
5.27 Organic Modeling	553
5.28 Understanding the Fluid Simulator	554
5.28.1 Understanding the Fluid Simulator	554
5.28.2 First of all	554
5.28.3 Setting up the scene	554
5.28.4 Setting up the simulation	555
5.28.5 Final notes	556
5.28.6 Extra Practice	556
5.28.7 Links	556
5.29 Creating a jewel in Blender	556
5.29.1 Modeling the jewel	556
5.29.2 Shading	560
5.29.3 The finishing touch	560
5.29.4 External links	560
5.30 Modeling a picture	560
5.30.1 Modeling a picture	560
5.30.2 Printing a Rendered Image	561
5.31 Modeling with the Spin Tool	561
5.32 Spin Tool Introduction	562
5.32.1 How the Spin Tool Works	562
5.32.2 Typical Work Flow	562
5.32.3 Post Modeling	566
5.33 Illustrative Example: Model a Wine Glass	566
5.33.1 Model half the outline	566
5.33.2 Spinning the outline	566
5.33.3 Cleaning up the Mesh	566
5.34 Creating Ogg-Theora movies using Blender	567
5.34.1 Converting saved frame picture files to Ogg Theora	567
5.34.2 Converting AVIs to Ogg Theora	567
5.35 Creating animated GIFs using Blender and Gimp	567
5.36 3D Tiling Backgrounds For The Web	568
5.36.1 Overview	568
5.36.2 Create The Object You Wish To Tile	568
5.36.3 Specify Your Tiling Area	568
5.36.4 Tile Away!	568
5.36.5 Camera Settings	570
5.36.6 Lighting & Materials	570
5.36.7 Rendering	570
5.36.8 'Shopping	571
5.37 Cool Things That Aren't That Obvious in Blender	572

5.37.1 Attribution	572
5.37.2 File Browser Functions	572
5.37.3 Object/Vertex Manipulation	572
5.37.4 Working with Meshes	575
5.37.5 Animation	576
5.37.6 UV Mapping, Particles and Texturing	576
5.37.7 Rendering	577
5.37.8 Miscellaneous	578
5.37.9 ???	579
5.37.10 Sculpt Mode Hotkeys	579
5.38 Troubleshooting	579
5.39 Creating Blender Libraries	580
5.39.1 How to Make a Library	580
5.39.2 Library Usefulness Checklist	580
5.39.3 Publishing Your Library	581
5.39.4 Beyond Libraries	581
5.39.5 Thank You	581
5.40 Add some depth with stereo	582
5.40.1 The stereo camera	582
5.40.2 Stereo viewing with the rig	583
5.40.3 Stereo rendering	583
5.41 Ways to create a “fluffy” effect (materials and lights)	585
5.41.1 Spherical Blend Texture	585
5.41.2 Backlighting	585
5.42 Human Body	586
5.43 Rendering Information	588
5.44 Using Blender Libraries	588
5.44.1 Indirect linking	588
5.44.2 Groups	588
5.44.3 See also	588
5.45 Beginning Modeling Final Project	588
5.46 Using Inkscape to make advanced Bezier curves	589
5.46.1 Introduction to Inkscape	589
5.46.2 Getting Started	589
5.47 Light Mapping	591
5.47.1 Needs Expert	591
5.48 Platonic Solids	591
5.48.1 External Link	591
5.49 Polygonal Modeling	591
5.50 Box Modeling	592
5.50.1 Before We Start	592

5.50.2	The Work Flow	592
5.51	Illustrative example: Model a Chair (Swan Chair)	592
5.52	Model a Chair-Preparations	593
5.52.1	Knowing our model	593
5.52.2	Loop study	593
5.53	Model a Chair-The Seat	593
5.53.1	Start with a Primitive	593
5.53.2	Rough it out	593
5.53.3	Adding More Details	594
5.54	Model a Chair-The Feet	594
5.54.1	Starting Primitive	594
5.54.2	The Upper Telescope	594
5.54.3	The Lower Telescope	595
5.54.4	The Stops	595
5.54.5	The Whole Chair	596
5.55	Illustrative Example: Modeling a Simple Human Character	596
5.56	Modeling a Human Character - Preparations	596
5.57	Modeling a Human Character - Modeling	596
5.58	Blocking the Figure	596
5.59	Detailing and Finishing	596
5.60	Polygon by Polygon modeling	596
5.60.1	Before we start	596
5.60.2	The Workflow	596
5.61	Animation Notes and FAQ	597
5.61.1	Blender 3D: Animation Notes and FAQ	597
5.61.2	IMPORTANT	597
5.61.3	Authors & Contributors	597
5.61.4	Introduction	597
5.61.5	Todo List	597
5.61.6	Animation FAQ	597
5.61.7	Armatures	597
5.61.8	Hotkeys	598
5.61.9	Approaches, Techniques, Tips and Tricks	598
5.61.10	Problems and Questions	599
5.61.11	General Armature Information	601
5.61.12	Building Armatures	602
5.61.13	Useful Rigs	602
5.62	Customization	603
5.62.1	Changing the Theme	603
5.63	Mist - Make Objects Opaque	603
6	Text and image sources, contributors, and licenses	606

6.1	Text	606
6.2	Images	619
6.3	Content license	681

0.1 Blender 3D: Noob to Pro



Blender WikiBookCover

0.1.1 About This Book

Blender 3D: Noob to Pro is a product of shared effort by numerous **team members** and anonymous editors. Its purpose is to teach people how to create three-dimensional computer graphics using Blender, a free software application.

This book is intended to be used in conjunction with other on-line resources that complement it:

- Other Blender-related Wikibooks on topics such as scripting and creating games;
- The Blender Wiki for technical documentation;
- User forums, such as the Blender Artists Forum.

While you can learn simply by reading the book, you'll get more out of the tutorials if you follow along. In order to do this, you'll need access to a computer with Blender installed. You can download Blender from the Blender Foundation's website; more detailed instructions are in the first module.

Version-specific content should be tagged with a note that looks like this:

0.1.2 Overview

The core of this book is a series of tutorials that increase in complexity, with later tutorials building on the preceding ones. While experienced users can skip ahead, *beginners are urged to proceed through the tutorials in sequence*.

The tutorials in the core series are grouped into four units:

1. *Background* — a basic orientation regarding:
 - computer graphics
 - the Blender user interface (UI)
2. *Basic Modeling and Shading* — basic techniques for building and rendering 3D models
3. *Broadening Horizons*
 - alternative modeling and rendering techniques
 - introductions to lighting, animation, and game creation
4. *Taking Off*
 - scripting
 - advanced techniques for modeling, animation and game creation

Each unit is subdivided into **sections**, which are made up of **modules**.

Three appendices are also provided:

- *Reference Material* — including:
 - Frequently-Asked Questions
 - Glossary
- *General Advice* — tips to help you get the most out of Blender
- *Miscellaneous Tutorials* — tutorials that aren't part of the core series

0.1.3 Table of Contents

Unit 1: Background

- Knowing before Making ⇡ **START HERE**
- What Blender Can Do
- Section 1A: 3D Concepts
 - 3D Geometry
 - Coordinate Transformations
 - Orthographic Views
 - Perspective Views

- Coordinate Spaces in Blender
- Section 1B: User Interface (UI)
 - Overview
 - Keystroke, Button, and Menu Notation
 - Non-standard Input Devices
 - Operating System-specific Issues
 - Blender User Interface
 - Blender Windowing System
 - User Preferences Windows
 - Properties Window
 - 3D View Windows
 - Object Mode
- Beyond Basics
 - Blender Memory Management
 - Using Bones
 - Mountains out of Molehills
 - Modeling a Volcano
 - Penguins from Spheres
 - Dicing With Depth (Dice Modeling)
 - Model a Goblet
 - Model a Silver Goblet
 - Model a Silver Goblet Another Way
 - Spin a goblet
 - Light a Silver Goblet (Early look at lighting)
 - Simple Vehicle
 - Simple Vehicle: Wheel tutorial 1
 - Simple Vehicle: Wheel tutorial 2
 - Simple Vehicle: Seat
 - Simple Vehicle: Rocket Launcher
 - Simple Vehicle: Body
 - Simple Vehicle: Some Assembly Required
 - Modeling a 3D Parachute in Blender
 - Model a Low Poly Head
 - Building a House
 - Pipe joints
 - Lighting Suzanne: Introductory one lamp lighting
- Curve and Path Modeling
 - Overview
 - Intro to Bézier Curves
 - Bevelling a Curve
 - NURBS Patches
 - Deforming Meshes using the Curve Modifier
- The Empty Object
- Using Reference Photos
 - Background Images
 - Aligning Vertices with a Guide Image
 - Modeling a Wolf from Guide Images
 - Using Bézier Curve to Model a 3D logo from a 2D logo
- Further Materials and Textures
 - Subsurface Scattering
 - Ray Tracing

- Using Textures
- Using a texture to make a material partially transparent
- Creating Basic Seawater
- Mountains Out Of Molehills 2
- Basic Carpet Texture
- The Rusty Ball
- Creating Pixar-looking eyes
- Procedural Eyeball
- Putting It All Together: A Dragon!
- Fire
- Fur
- Fireworks
- Particles forming Shapes
- Billboard Animation
- Soft Bodies
- Soft Body Animation
- Simple Cloth Animation
- Soft Body with Wind
- Blender Game Engine
- Blender Game Engine Basics- Rolling Ball
- Platformer: Controls and Movement
- Maze: Force and Multiple Levels
- Platformer: Improving the Physics
- How to Make an Executable
- Build a Skybox
- Basic Mouse Pointer
- Text in BGE
- Platformer: Creating the Engine with Python

Unit 3: Broadening Horizons

- The UV/Image Editor
- UV Maps - Pasting photos to 3D surfaces
 - UV Map Basics
 - Realistic Eyes In Blender
- Lighting
 - Beginning Lighting
 - Understanding Real Lights
 - Understanding Blender Lights
 - Basic Lighting Rigs
 - Faked Global Illumination with Blender internal
- Practicing Good Parenting
- Basic Animation
 - Overview
 - Introduction to Keyframing
 - The Ways of the Animator
 - Animation Editors
 - Introducing the Graph Editor
 - Animation Rendering
 - Lattice Modifier
 - Bouncing Ball with Lattice
 - Creating Basic Water Animation
 - Flying Through a Canyon
 - Using the Sequencer to Compile Frames into an Animation
- Further Rendering Options
- Particle Systems
 - Overview

Unit 4: Taking Off with Advanced Tutorials

- Python Scripting
 - Introduction
 - Anatomy Of An Addon
 - A User Interface For Your Addon
 - Adding A Custom Property
 - A Separately Installable Addon
 - Object, Action, Settings
- Advanced Modeling
 - Overview
 - High Dynamic Range imaging (HDRi)
 - Creating a Light Probe
 - Landscape Modeling with Heightmaps
 - How to Do Procedural Landscape Modeling
 - Landscape Modeling I: Basic Terrain
 - Landscape Modeling II: Texture Stenciling
 - Landscape Modeling III: Exporting as a Heightmap
 - Advanced Materials and Textures

- Bump Mapping
- Normal Mapping
 - Texture Normals
 - Color Map Normals
- Nodes
 - Introduction
 - Texture Nodes
 - Material Nodes
 - Compositing
 - Further Compositing: A Portal Effect
- Advanced Rendering
 - Introduction to Cycles
 - A Glass Material in Cycles
 - Dealing with Fireflies in Cycles
 - Fireflies in Cycles, Continued
 - Procedural Eyeball in Cycles
 - Introduction to Freestyle
- Advanced Animation
 - Overview
 - Introduction
 - Guided Tour:
 - Armature Object
 - Armature Object in Object Mode
 - Armature Object in Edit Mode
 - Armature Object in Pose mode
 - Mesh Object
 - Connection between Armature and Mesh
 - Envelope
 - Vertex Groups & Weight Paint
 - Shape Keys
 - Lip-Sync with Shape Keys
 - Constraints
 - Copy Location
 - Copy Rotation
 - Track-To
 - Floor
 - Locked Track
 - Follow Path
 - Stretch-To
 - IK Solver
 - Timeline Window
 - Graph Editor
 - Dope Sheet
 - NLA Editor
- Introduction To NLA Editor
- Working Example: Bob
 - Building the Rig
 - Deform the Mesh
 - Create a Walk Cycle
- Working example: Piston, Rod and Crank
- Working example: Cutting Through Steel
- Advanced Game Engine
 - Overview
 - Advanced Game Engine Techniques (GUI)
 - Creating Pop-Up Menus
 - Creating Dropping Menus
 - The “5-Layer” Button
 - Creating Object Outlines
 - Advanced Game Engine Techniques (Python)
- Hacking Blender
 - Introduction to Game Engine Source

Appendices

Reference Material

- Glossary
- Frequently Asked Questions
- Tutorial Links List
- Hotkeys
- Output Formats
- Image Portfolio
- Blender Glossary
- Materials Directory: Every Material Known To Man
- Sources of free 3D models - Sources of free 3D models for additional study
- All Blueprints Links – blueprints from all over the Web
- Materials, Textures, Photos - Sources of free materials, textures and photos

Advice General advice:

- Asking for Help
- Tips for a Successful Project
- Modeling Realistically
- Modeling tips

Performance tips (for making Blender run faster):

- Cheat the 3D
- Performance vs. Quality

Miscellaneous Tutorials *This is our attic, mostly tutorials that could be useful to some extent if they would be revamped completely, but are of little use at the moment. If you can contribute to some of them, go ahead and rewrite them to your liking!*

- Modeling a Gingerbread Man
- Modeling a simple space-ship
- Create an animated GIF wallpaper (Blender/GIMP)
 - Part 1 - Preparing the Scene
- Creating Weapons based on 2D Images
- Modeling with Meta Balls
- Match Moving
- Match Moving/Motion Tracking with Icarus and Blender
- Create a Clayman
- Organic Modeling
- Understanding the Fluid Simulator
- Creating a jewel in Blender
- Modeling a picture ■
- Modeling with the Spin Tool
 - Spin Tool Introduction
 - Illustrative Example: Model a Wine Glass
- Creating Ogg-Theora movies using Blender
- Creating animated GIFs using Blender and Gimp
- 3D Tiling Backgrounds For The Web
- Cool Things That Aren't That Obvious in Blender
- Troubleshooting Common Technical Issues and What to do About Them

- Creating Blender Libraries
- Add some depth with stereo
- Ways to create a “fluffy” effect (materials and lights)
- Human Body
- Rendering Information
- Using Blender Libraries
- Beginning Modeling Final Project
- Using Inkscape to make advanced Bezier curves
- Light Mapping
- Platonic Solids
- Polygonal Modeling
 - Box Modeling
 - Illustrative example: Model a Chair (Swan Chair)
 - Model a Chair-Preparations
 - Model a Chair-The Seat
 - Model a Chair-The Feet
 - Illustrative Example: Modeling a Simple Human Character
 - Modeling a Human Character - Preparations
 - Modeling a Human Character - Modeling
- Polygon by Polygon modeling
- Animation Notes and FAQ
- Customization
- Mist - Make Objects Opaque

Additional Resources

- Blender Wiki PDF Manual (1,426 pages)

Chapter 1

Unit 1 - Background

1.1 Knowing before Making

Blender is a powerful and complex 3D modeling and rendering package. Before you can use it effectively to *make* things, you need to *know* a few things about how it works:

- the process of 3D modeling and rendering (what Blender does)
- some rudiments of 3D analytic geometry (axes and coordinates)
- orthographic and perspective views of a 3D object
- local coordinate systems and child objects
- the fundamentals of Blender's user interface (hotkeys, windows, and menus)
- how to view a 3D scene from different vantage points (in Blender)

This unit is devoted entirely to this sort of background knowledge. You won't create your first Blender model until the next unit.

Knowing this, you might be tempted to skip ahead. Depending on your background, that may or may not work. For instance, if you've used other 3D graphics packages, you might be able to skim (or skip) ahead as far as the user interface tutorial. But if there's any doubt, please proceed through the tutorials in sequence.

Blender is not the kind of software you can launch into and grope about until you find your way. It's not like exploring an unfamiliar city. It's more like flying a spaceship. If you hop into the pilot's seat without knowing the fundamentals, you'll be lucky to ever get off the ground, and it'd take a miracle for you to reach your destination safely.

A Word about Jargon

Like any subject, 3D modeling has its own jargon: terminology specific to the subject and ordinary words that have special meanings in the context of computer graphics.

In this book, important new words are highlighted on first appearance and defined soon after. If you suspect you've missed (or forgotten) the meaning of a word, try looking it up in the [Glossary](#).

Things You'll Need

In order to work through the tutorials, you'll need access to a computer that has Blender installed (download the latest [stable release](#)).

Depending on what is installed on your system, you may also need the appropriate Python installation. Each version of Blender works with only one specific version of Python, which is generally included in the download.

Installation instructions

Since Blender is open-source software, you can download the source code and build it yourself, but it's easier to download a pre-built install package. Install packages are provided for each supported operating system:

- Microsoft Windows (32-bit and 64-bit)
- Linux (32-bit and 64-bit x86)
- Apple Mac OS X (PowerPC and Intel)
- FreeBSD (32-bit and 64 bit)
- Solaris
- Irix

Many Linux distributions have Blender available in their package repositories, though it may be a slightly older version. You can use your system's package manager to download and install the package.

Windows users get to choose between an executable installer ("setup wizard") and a ZIP archive.

After the installation process is finished, Blender should appear in the Graphics section of your desktop environment application menu.

You may also want to download a 2D image editor, such as GIMP (), Paint.NET (), or Photoshop. For viewing video files, you may want to install VLC () media player.

It's a good idea to have pencil and paper handy for sketching and taking notes. There's a lot to absorb. Taking notes as you go will pay dividends later.

Where to Go for Help

If you get stuck, you can ask for help from other Blender users in the appendices.



“A Lonely House”, by Maygel

1.1.1 Additional Resources

Many modules have a section like this at the bottom, listing websites with information on the topics covered in the module.

- Blender system requirements
- the *GIMP* wikibook
- Windows installation
- Linux installation
- OSX installation
- About the Python programming language

1.2 What Blender Can Do

In this module, you'll learn what Blender does, both in terms of the product (images) and the process (3D modeling).

Blender is a **free software** package for authoring “three-dimensional” (3D) graphics (also known as *computer graphics* or “CG”), including still images, games, and video.

While the end-product of most Blender projects is a two-dimensional (2D) raster image on a flat surface (be it a monitor, movie screen, or sheet of paper) except for Head Mounted Virtual Reality applications, the images are said to be “3D” because they exhibit the illusion of depth. In other words, someone looking at the image can easily tell which parts are meant to be closer and which are farther away.

1.2.1 An Example

Here's a realistic still image that was authored with Blender.

Look closely at the building.

- Because it is obscured by the building, you can tell that the tree-lined hillside is *behind* the building instead of vice versa.
- The way the top and bottom edges of the front wall appear to converge toward the base of the tree allow you to judge the angle between the front wall and your viewpoint.
- Your brain interprets dark portions of the wall as shadows, allowing you to estimate where the light is coming from, even though the sun is outside the frame of the image.

While an illusion of depth can be authored by hand with 2D graphics software (or a paintbrush!), Blender provides a much easier way.

It's likely that the lonely house never existed outside of the artist's mind. Instead of building a big set on a rural lot in Germany, waiting for the right light, and photographing it, the author built a scene in a virtual 3D world—one contained inside a computer. This is called CGI (Computer Generated Imagery). He or she then used Blender to render the scene (convert it into a 2D image). You can view more of what Blender can do at the Blender gallery: <http://www.blender.org/features/>

1.2.2 Steps in the 3D Production Process

To produce an image like the one above involves two major steps to start with:

- Modelling, which is the creation of your miniature 3D world, also known as a model or scene. This involves defining the geometry of the objects, making it look like they are made out of particular materials, setting up the lighting, and defining a camera viewpoint.
- Rendering, which is the actual generation of the image of the world from the viewpoint of the camera (taking a “photograph” of the scene, if you like), for your audience to enjoy.

3D is often used to produce not just single still images, but animations as well. This requires some additional steps:

- Rigging — setting up a rig, namely a way of deforming (changing the shape of) a character in various repeatable ways to convincingly mimic joint movements, facial expressions and other such actions of real-life people or animals.
- Posing — choreographing the positions of the objects and their parts in the 3D scene over time, using the previously-created animation rigs
- Rendering now involves creating a whole sequence of frames representing movement over time, rather than just a single still frame.

But that's not all. There are frequently additional processes to embellish the results of the above, to make them look more realistic:

- Sculpting — a more organic form of modelling objects by shaping them as though they were made out of clay. This produces more complicated, irregular shapes which mimic real objects found in nature, as opposed to clean, simple, geometrical ones which mostly only exist in the world of mathematics.
- Texture painting — You're probably familiar with programs that let you paint an image on a 2D digital canvas. Such programs are commonly used in 3D production, to create textures which are “wrapped” around the surfaces of 3D objects to give them a more interesting appearance. 3D programs also often allow direct painting on the surfaces of those objects, so the effect of the design can be observed immediately, instead of having to go through a separate paint-on-a-flat-surface-then-wrap sequence of steps.
- Physical modelling — simulating the behaviour of real-world objects subject to real-world forces, for example hard balls colliding, soft cloth draping itself over an obstacle under gravity, water flowing and pouring. Mathematical formulas are available for these that give results very close to real life, all you need is the computing power to calculate them.
- Motion capture, or mocap: producing convincing animations, particularly ones that look like the movements of real people (walking, running, dancing etc) can be hard. Hence the technique of capturing the motions of live actors, by filming them with special markers attached to strategic points on their bodies, and doing computer processing to track the movements of these markers and convert them to corresponding movements of an animation rig.
- Compositing — this is where 3D renders are merged together with real photographic/live-action footage,

to make it look like a rendered model is in the middle of a real-world scene, or conversely a real live actor is in the middle of a rendered scene. If done with proper skill, in particular due care to matching the effects of lights and shadows, the viewer becomes unable to tell what is real and what is not!

And just to add another complication to the mix, there are two kinds of rendering:

- Real-time rendering is rendering that has to happen under tight time constraints, typically for interactive applications like video gaming. For example, most gamers nowadays consider that the screen has to be updated 60 times per second in order to render smooth motion and respond quickly enough to player actions. These time constraints impose major limitations on the kinds of rendering techniques that can be used.
- Non-real-time rendering is where the time constraints are not so tight, and quality is the overriding factor. For example, when producing a single still frame, it may not matter so much that it takes minutes or hours to do so, because the beauty and detail of the final image is worth it. When rendering a Hollywood-quality movie, it may still take hours per frame, but the use of a renderfarm of hundreds or thousands of machines, all working on different frames at the same time, allows the entire sequence to complete in just a few weeks.

But wait, there's more: There are also some areas, which might be considered to be stepping outside of traditional 3D production work, where Blender provides functionality:

- **Video editing** — having rendered your animation sequences and shot your live-action footage, you will want to combine them in a properly-timed linear sequence to tell a coherent story.
- **3D printing** — Though still in its early days yet, many people are already experimenting with creating objects using 3D printers. The shape data may be obtained from real objects with 3D scanning, or it may be created from scratch using 3D modelling, or you can even combine both processes.

Blender is a capable tool for every single one of these processes. There's quite a lot there, isn't there? But don't be too intimidated: this Wikibook will take things step by step, and you will be able to produce some fun stuff from early on.

1.2.3 Additional Resources

-

-  3D rendering at Wikipedia.
-
-  Comparison of 3D computer graphics software at Wikipedia.
-
-  Computer-generated imagery at Wikipedia.
-
-  Depth perception at Wikipedia.
- Blender Art Gallery
- Blender Homepage

1.3 3D Geometry

If you haven't previously studied 3D graphics, technical drawing, or analytic geometry, you are about to learn a new way of visualizing the world, an ability that's fundamental to working with Blender or any 3D modeling tool.

3D modeling is based on geometry, the branch of mathematics concerned with spatial relationships, specifically analytical geometry, which expresses these relationships in terms of algebraic formulas. If you have studied geometry, some of the terminology will be familiar.

1.3.1 Coordinates And Coordinate Systems

Look around the room you're in. The odds are it will have a cuboidal shape, with four vertical walls at right angles to each other, a flat, horizontal floor, and a flat, horizontal ceiling.

Now imagine there's a fly buzzing around the room. The fly is moving in three-dimensional space. In mathematical terms, that means its position within the room at any given moment, can be expressed in terms of a unique combination of three numbers.

There are an infinite number of ways —*coordinate systems*— in which we could come up with a convention for defining and measuring these numbers, i.e. the *coordinates*. Each convention will yield different values even if the fly is in the same position. Coordinates only make sense with reference to a specific coordinate system! To narrow down the possibilities (in a purely arbitrary fashion), let us label the walls of the room with the points of the compass: in a clockwise direction, North, East, South and West. (If you know which way really is north, feel free to use that to label the walls of your room. Otherwise, choose any wall you like as north.)

Consider the point at floor level in the south-west corner of the room. We will call this (arbitrary) point the *origin* of our coordinate system, and the three numbers at this point will be $(0, 0, 0)$. The first of the three numbers will be the distance (in some suitable units, let's say meters) eastwards from the west wall, the second number will be the distance north from the south wall, and the third number will be the height above the floor.

Each of these directions is called an *axis* (plural: *axes*), and they are conventionally labelled X, Y and Z, in that order. With a little bit of thought, you should be able to convince yourself that every point within the space of your room corresponds to exactly one set of (x, y, z) values. And conversely that every possible combination of (x, y, z) values, with $0 \leq x \leq W$, $0 \leq y \leq L$ and $0 \leq z \leq H$ (where W is the east-west dimension of your room, L is its north-south dimension, and H is the height between ceiling and floor) corresponds to a point in the room.

The following diagram illustrates how the coordinates are built up, using the same colour codes that Blender uses to label its axes: red for X, green for Y and blue for Z (an easy way to remember this if you're familiar with RGB is the order -- Red X, Green Y, Blue Z). In the second picture, the x value defines a plane parallel to the west wall of the room. In the third picture, the y value defines a plane parallel to the south wall, and in the fourth picture, the z value defines a plane parallel to the floor. Put the planes together in the fifth picture, and they intersect at a unique point.



This style of coordinate system, with the numbers corresponding to distances along perpendicular axes, is called *Cartesian coordinates*, named after René Descartes, the 17th-century mathematician who first introduced the concept. Legend has it that he came up with the idea after watching a fly buzzing around his bedroom!

There are other ways to define coordinate systems, for example by substituting direction angles in place of one or two of the distance measurements. These can be useful in certain situations, but usually all coordinate systems in Blender are Cartesian. However, in Blender, switching between these coordinate systems is simple and easy to do.

Negative Coordinates

Can coordinate values be negative? Depending on the situation, yes. Here we are only considering points within our room. But suppose instead of placing our origin in the bottom southwest corner, we put it in the middle of the room, halfway between the floor and ceiling. (After all,

it is an arbitrary point, we can place it wherever we like, as long as we agree on its location.) If the X-coordinate is the distance *east* from the origin, how do we define a point *west* of the origin? We simply give it a negative X-coordinate. Similarly, points north of the origin have a positive Y-coordinate, those south of it, have negative Y-coordinates. Points above the origin have a positive Z-coordinate, those below it, a negative Z-coordinate.

Handedness Of Coordinate Systems

It is conventional for most Cartesian coordinate systems to be *right-handed*. To understand this, hold the thumb, index finger and middle finger of your right hand perpendicular to each other:

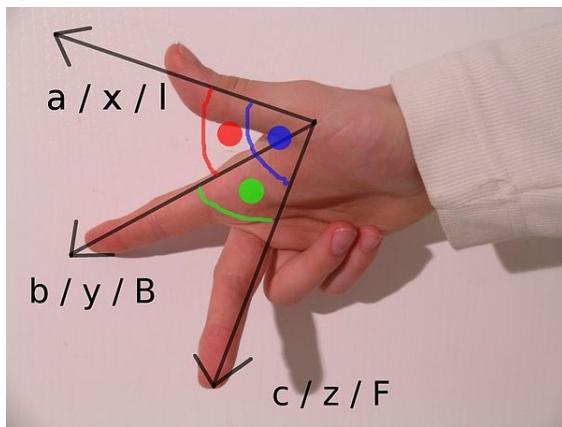


Figure 1: The three axes form a right-handed system

Now orient your hand so your thumb points along the X-axis in the positive direction (direction of increasing coordinate numbers), your index finger along the positive Y-axis, and your middle finger along the positive Z-axis. Another way of looking at it is, if you placed your eye at the origin, and you could see the three arrows pointing in the directions of positive X, positive Y and positive Z as in Figure 1, the order X, Y, Z would go clockwise.

Another way to visualize this is to make a fist with your right hand, with your curled fingers towards you. Stick out your thumb directly to the right (X). Now aim your pointer finger straight up (Y). Finally, make your middle finger point toward yourself (Z). This is the view from directly above the origin.

1.3.2 Axes Of Rotation

Consider a spinning sphere. Every point on it is moving, except the ones along the axis. These form a motionless line around which the rest of the sphere spins. This line is called the *axis of rotation*.

More precisely, the axis of rotation is a point or a line connecting points that do not change position while that

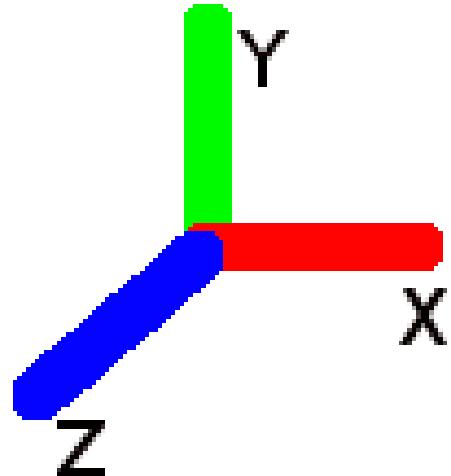
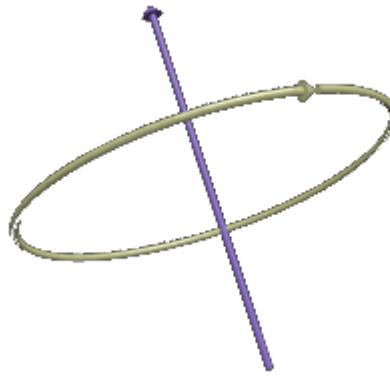


Figure 2: Another view of right-handed system

object rotates, drawn when the observer assumes he/she does not change position relative to that object over time.

Conventionally, the *direction* of the axis of rotation is such that if you look in that direction, the rotation appears clockwise, as illustrated below, where the yellow arrow shows the rotational movement, while the purple one shows the rotation axis:



To remember this convention, hold your right hand in a



thumbs-up gesture:

If the rotation

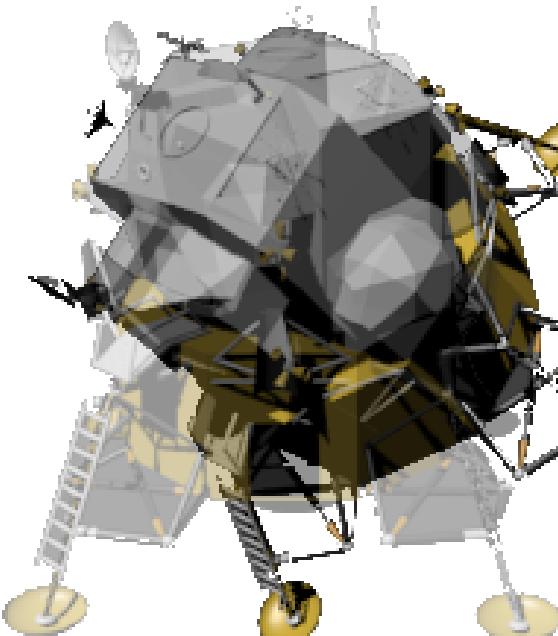
follows the direction of your curled fingers, then the direction of the axis of rotation is considered to be the same as the direction which the thumb is pointing in.

This gesture is a different form of the right-hand rule and is sometimes called *the right-hand grip rule*, *the corkscrew-rule* or *the right-hand thumb rule*. From now on we will refer to it as 'the right-hand grip rule'.

When describing the direction of a rotating object, do not say that it rotates *left-to-right/clockwise*, or *right-to-left/counterclockwise*. Each of these on their own are meaningless, because they're relative to the observer. Instead of saying this, find the direction of the axis of rotation and draw an arrow to represent it. Those who know the right-hand grip rule will be able to figure out what the direction of rotation of the object is, by using the rule when interpreting your drawing.

1.3.3 Additional Resources

- the [Geometry](#) wikibook
-
-
-  [Analytic geometry](#) at Wikipedia.
-
-  [Cartesian coordinate system](#) at Wikipedia.
-
-
-  [Right-hand rule](#) at Wikipedia.
-
-
-  [Rotation](#) at Wikipedia.



Object rotated 45°

1.4 Coordinate Transformations

1.4.1 Coordinate Transformations

A *transformation* is any operation that changes coordinate values in some way. For example, if you pick up an object and move it to a different place in the room without changing its orientation, then the coordinates of each point on the object relative to the room are adjusted by an amount that depends on the distance and direction between the old and new positions. This is called a *translation* transformation.

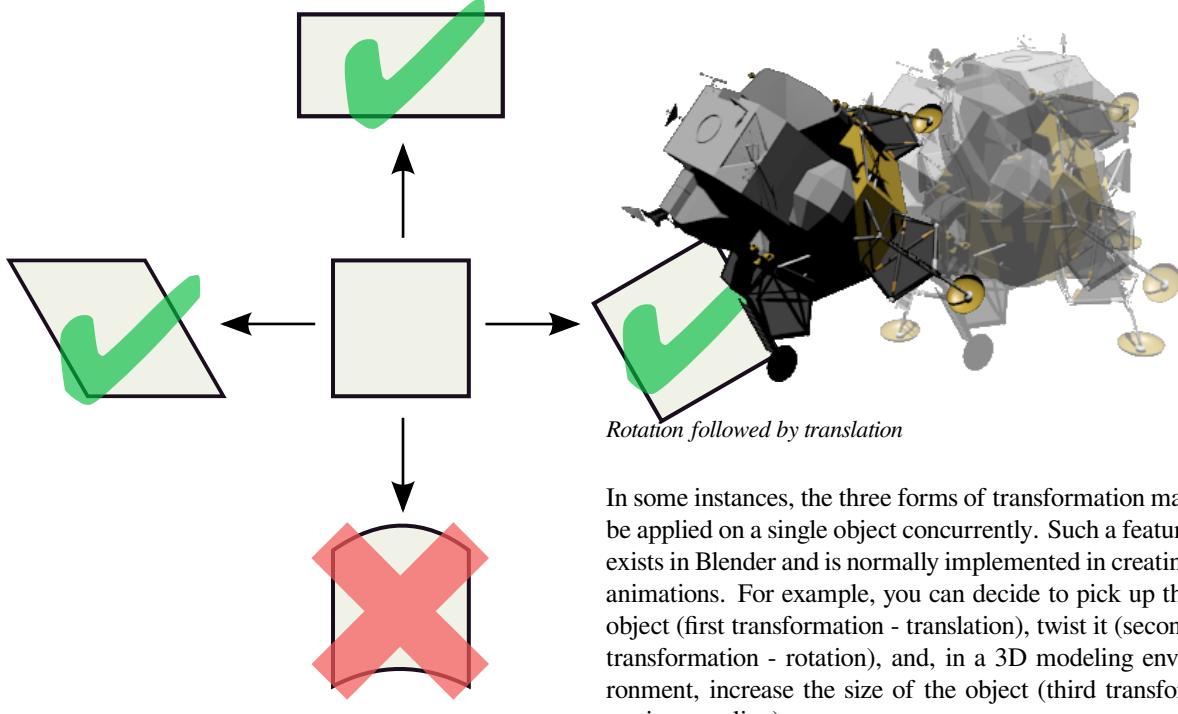
Simply turning the object without moving it from its original location is called *rotation*.

If the object were to get bigger or smaller, that is a *scaling* transformation. In the real world, only a few objects can be scaled in this way. For example, a balloon can be inflated or deflated to a larger or smaller size, but a bowling ball cannot. Regardless of what can and can't be re-sized in the real world, any object can be scaled (re-sized) in the world of computer graphics. Scaling may be *uniform*, i.e. apply equally in all dimensions, or *non-uniform*.

1.4.2 Linear Transformations

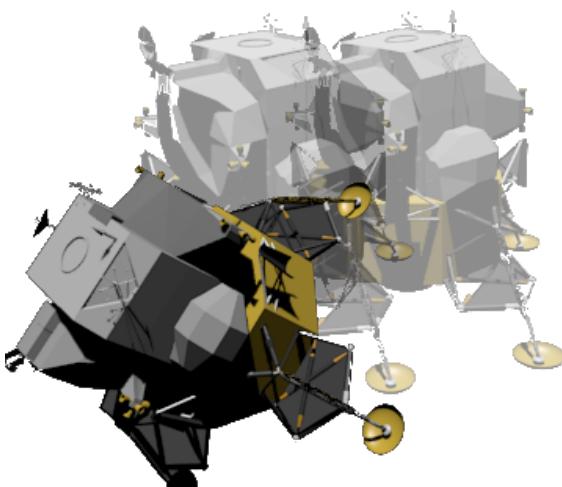
The main types of coordinate transformations we're concerned with are called *linear transformations*. Lines that were straight before the transformation remain straight. i.e. they do not become curved. For example, the following diagram illustrates three linear transformations applied to the square in the center: Clockwise from the left, a *shear* or *skew*, a scale, and a rotation, plus one non-linear transformation that causes two sides of the box to become curved.

versus the result of doing the rotation first:



1.4.3 Multiple Transformations

It is possible to *concatenate* or *compose* a series of transformations. The resulting transformation can do many things in one operation — translation, rotation, scaling etc. However, the order of composition of the component transformations becomes important. In general, transformations are not *commutative*. For example, compare the result of moving our model some distance along the Y axis followed by rotating it about the X axis (If this doesn't make sense, consider that the axes are fixed, they aren't moving with the object. More on that later [Global and local coordinates](#)):



Translation followed by rotation

In some instances, the three forms of transformation may be applied on a single object concurrently. Such a feature exists in Blender and is normally implemented in creating animations. For example, you can decide to pick up the object (first transformation - translation), twist it (second transformation - rotation), and, in a 3D modeling environment, increase the size of the object (third transformation - scaling).

1.4.4 Inverse Transformations

Often there is a need to find the *inverse* of a transformation. That is, a transformation that has the opposite effect. For example, a rotation of $+45^\circ$ about the X axis is undone by a rotation of -45° around the same axis.

Inverses have many uses, one of which is to simplify the construction of certain kinds of transformations.

For example, it is easy to construct a rotation transformation about the X, Y or Z-axis of the coordinate system. But what about a rotation of Θ° around an arbitrary axis? This can be made out of the following parts:

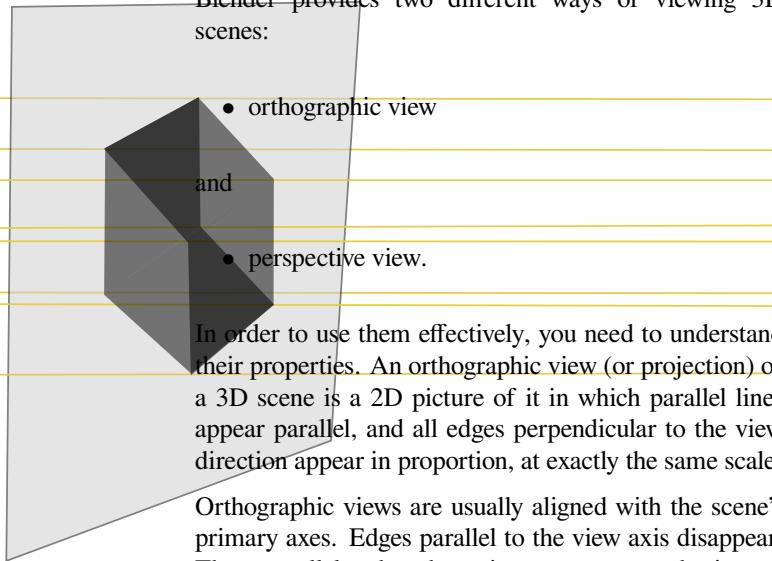
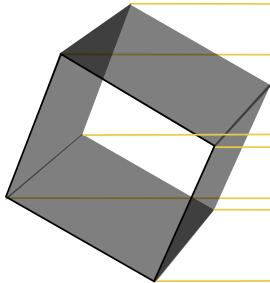
- a translation that makes the rotation axis pass through the origin.
- rotations about the Y and/or Z axes, as appropriate, so the rotation axis lies along the X axis.
- a rotation of Θ° about the X axis.
- the inverse of the rotations that aligned the rotation axis with the X axis.
- the inverse of the translation that made the rotation axis pass through the origin.

Most of the transformations we deal with in 3D modelling have an inverse, but not all. See the next section for some that don't.

1.4.5 Projections

Most of our display and output devices are not three-dimensional. Thus, three-dimensional images need to be *projected* onto a two-dimensional surface (like a display screen or a printed page) before we can see them.

There are two main ways to perform such projections. One is *orthographic* projection, where parallel lines are drawn from all points of the three-dimensional object until they intersect a plane representing the display surface:



In order to use them effectively, you need to understand their properties. An *orthographic view* (or *projection*) of a 3D scene is a 2D picture of it in which parallel lines appear parallel, and all edges perpendicular to the view direction appear in proportion, at exactly the same scale.

Orthographic views are usually aligned with the scene's primary axes. Edges parallel to the view axis disappear. Those parallel to the other primary axes appear horizontal or vertical. The commonly used orthographic views are front, side, and top views, though back and bottom views are possible.

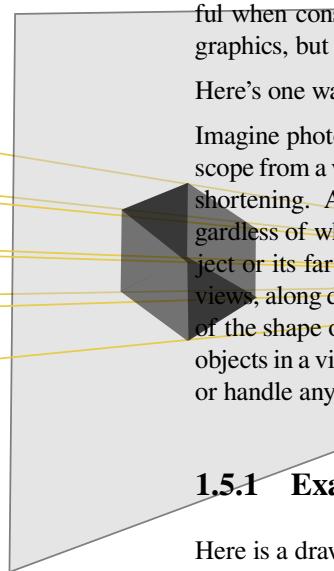
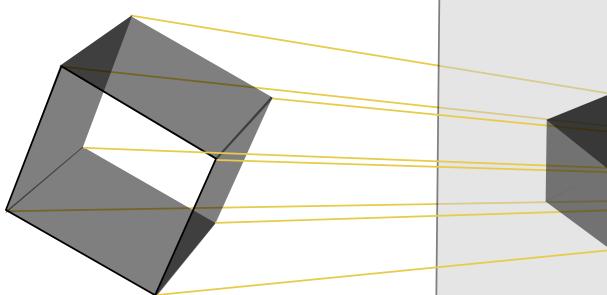
Uniform scale makes an orthographic view very useful when constructing 3D objects, not only in computer graphics, but also in manufacturing and architecture.

Here's one way to think about the orthographic view:

Imagine photographing a small 3D object through a telescope from a very great distance. There would be no foreshortening. All features would be at the same scale, regardless of whether they were on the near side of the object or its far side. Given two (or preferably three) such views, along different axes, you could get an accurate idea of the shape of the object, useful for "getting the feel" of objects in a virtual 3D world where you're unable to touch or handle anything!

More on orthographic projections

The other way is *perspective* projection, where the lines drawn are not parallel, but intersect at a point representing the location of the eye of the viewer:



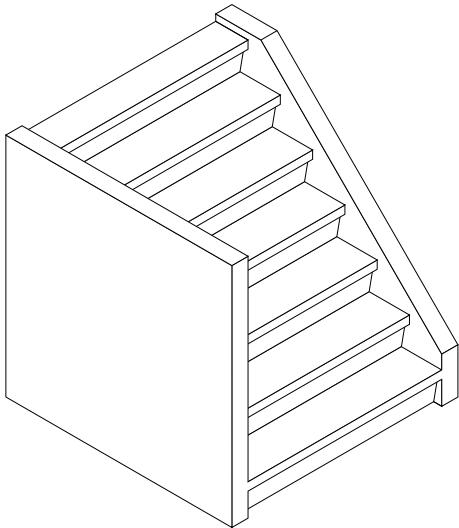
1.5.1 Example

Here is a drawing of a staircase:

and here are three orthographic views of the same staircase, each outlined in red:

The views are from the front, top, and left. Dashed lines represent edges that, in real life, would be hidden behind something, such as the left wall of the staircase. (Think of each view as an X-ray image.)

Projections are also linear transformations. But since they take a three-dimensional space and flatten it onto a two-dimensional surface, some information is lost. Those transformations are *non-invertible* i.e. they cannot be undone, at least in a unique way as the depth information is gone.



An isometric view of a staircase

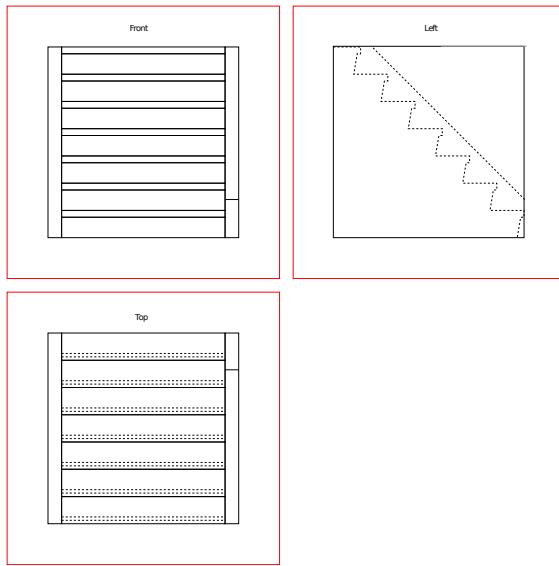


Figure 1: Orthographic views of a staircase

The leading edges of the steps are visible in both the front and top views. Note that they appear parallel and of equal length in 2D, just as they are in 3D reality.

1.5.2 Additional Resources

-
- [Orthographic projection at Wikipedia](#).

1.6 Perspective Views

As you know, the main reason for modeling 3D objects in Blender is to render images that exhibit the illusion of depth.

Orthographic views are great for building a house, but seriously flawed when it comes to creating realistic images of the house for use in a sales brochure. While a builder wants blueprints that are clear and accurate, a seller wants imagery that's aesthetically pleasing, with the illusion of depth. Blender makes it easy to use tricks like perspective, surface hiding, shading, and animation to achieve this illusion.

How does perspective work?

The essence of perspective is to represent parallel edges (in a 3D scene) by edges (in the 2D image) that *are not parallel*. When done correctly, this produces foreshortening (nearby objects are depicted larger than distant ones) and contributes to the illusion of depth.

Perspective is challenging to draw by hand, but Blender does it for you, provided you give it a 3D model of the scene and tell it where to view the scene from.

If you're confident you understand perspective, you can skip the rest of this module and proceed to the “Coordinate Spaces in Blender” module.

1.6.1 One-point Perspective

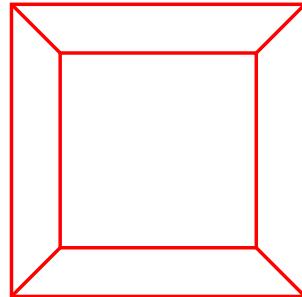


Figure 1: 1-Point Perspective.

Drawing classes teach various kinds of perspective drawing: one-point perspective, two-point perspective, and three-point perspective. In this context, the word “point” refers to what artists call the *vanishing point*.

When you're looking at a 3D object head-on and it's centered in your view, that is an example of one-point perspective.

Imagine looking down a straight and level set of train tracks. The tracks appear to converge at a point on the horizon. This is the vanishing point.

The image on the right is a 2D image of a cubic lattice or framework. Like any cube, it has six square faces and twelve straight edges. In the 3D world, four of the edges are parallel to our line-of-sight. They connect the four corners of the nearest square to the corresponding corners of the farthest one. Each of these edges is parallel to the other three.

In the 2D image, those same four edges appear to converge toward a vanishing point, contributing to the illusion of depth. Since this is one-point perspective, there is a single point of convergence at the center of the image.

1.6.2 Two-point Perspective

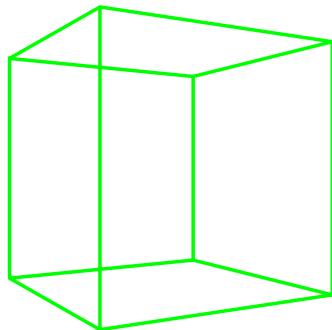
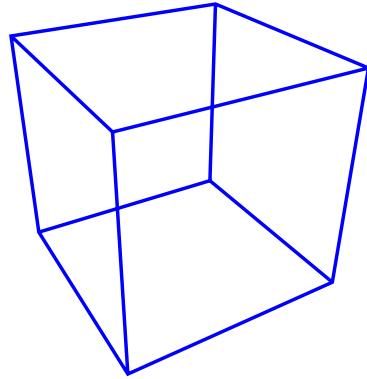


Figure 2: 2-Point Perspective.

Now the cube is at eye level, and you're near one of its edges. Since you're not viewing it face-on, you can't draw it realistically using one-point perspective. The horizontal edges on your left appear to converge at a point on the horizon to the left of the cube, while those on the right converge to the right. To illustrate the cube with a good illusion of depth, you need two vanishing points.

1.6.3 Three-point Perspective

Now imagine you're above the cube near one of its corners. To draw it, you'd need three vanishing points, one for each set of parallel edges.

From that perspective, there are no longer any edges which appear parallel. The four vertical edges, the four left-right edges, and the four in-out edges each converge toward a different vanishing point.

3-Point Perspective.

1.6.4 Additional Resources

- [Perspective \(graphical\)](#) at Wikipedia.

1.7 Coordinate Spaces in Blender

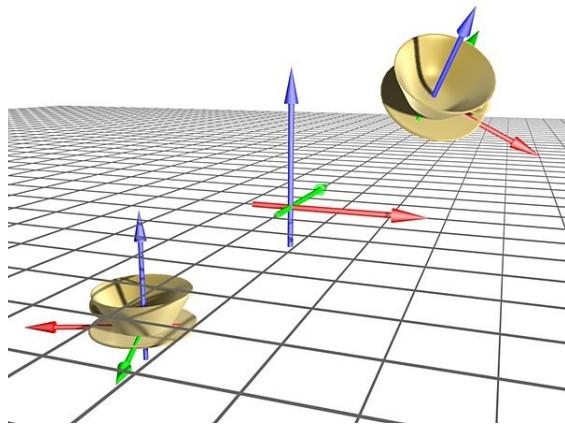


Figure 1: Objects in a three dimensional space. In the center of the coordinate system is the origin of the global coordinate system.

We'll start looking at how 3D scenes are represented in Blender.

As was explained in the “3D Geometry” module, Blender represents locations in a scene by their coordinates. The coordinates of a location consist of three numbers that define its distance and direction from a fixed origin. More precisely:

- The first (or x-) coordinate of the location is defined

as its distance from the YZ plane (the one containing both the Y and Z axes). Locations on the +X side of this plane are assigned positive x-coordinates, and those on the -X side are given negative ones.

- Its second (or y-) coordinate is its distance from the XZ plane, with locations on the -Y side of this plane having negative y-coordinates.
- Its third (or z-) coordinate is its distance from the XY plane, with locations on the -Z side of this plane having negative z-coordinates.

Thus the origin (which lies at the junction of all three axes and all three planes) has the coordinates (0, 0, 0).

1.7.1 Global and local coordinates

Blender refers to the coordinate system described above as the global coordinate system, though it's not truly global as each scene has its own global coordinate system. Each global coordinate system has a fixed origin and a fixed orientation, but we can view it from different angles by moving a virtual camera through the scene and/or rotating the camera.

Global coordinates are adequate for scenes containing a single fixed object and scenes in which each object is merely a single point in the scene. When dealing with objects that move around (or multiple objects with sizes and shapes), it's helpful to define a **local coordinate system** for each object, i.e. a coordinate system that can move with, and follow the object. The origin of an object's local coordinate system is often called the **center of the object** although it needn't coincide with the geometrical center of the object.

3D objects in Blender are largely described using vertices (points in the object, singular form: **vertex**). The global coordinates of a vertex depend on:

- the (x, y, z) coordinates of the vertex in the object's **local** coordinate system
- the location of the object's center
- any rotation (turning) of the local coordinates system relative to the global coordinate system, and
- any scaling (magnification or reduction) of the local coordinate system relative to the global coordinate system.

For example, the teacup in Figure 1 is described by a mesh model containing 171 vertices, each having a different set of local (x, y, z) coordinates relative to the cup's center. If you translate the cup (move it without rotating it), the only bits of the model that have to change are the global coordinates of the center. The local coordinates of all its vertices would remain the same.

Coordinates of child objects

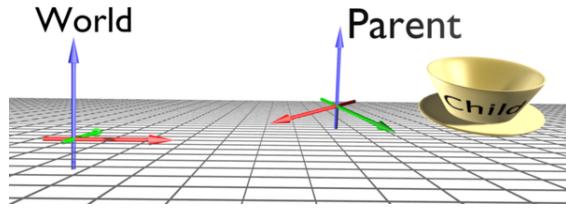


Figure 1b: A parent serves as the source of the global coordinates for its child object. The child is the cup; the parent's orientation is shown with the colored arrows.

Any object can act as a parent for one or more other objects in the same scene, which are then referred to as its children. (An object cannot have more than one direct parent, but parent objects may themselves be the children of other objects.)

If an object has a parent, its position, rotation, and scaling are measured in the parent's local coordinate system, almost as if it were a vertex of the parent. i.e. the position of the child's center is measured from the parent's center instead of the origin of the global coordinate system. So if you move a parent object, its children move too, even though the children's coordinates have not changed. The orientation and scaling of a child's local coordinate system are likewise measured relative to those of its parent. If you rotate the parent, the child will rotate (and perhaps revolve) around the same axis.

Parent-child relationships between objects make it simpler to perform (and animate) rotations, scaling and moving in arbitrary directions. In Fig. 1b the teacup is a child object of the coordinate cross on the right. That cross is itself the child of an invisible parent. (It is both a parent and child.) In the cup's local coordinate system, it is not rotating, but as the cross on the right rotates around its Z axis, it causes the cup to rotate and revolve. In real animations, it will be much easier when the character holding the cup rotates, the cup changes its position respectively.

1.7.2 View coordinates

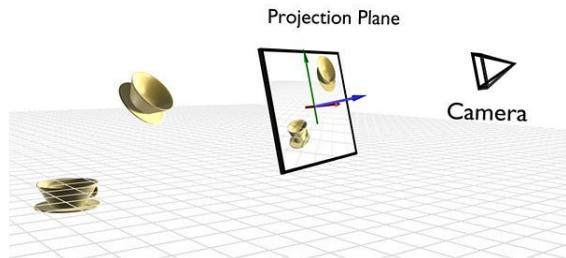


Figure 2: View coordinates and Projection Plane

Taking the viewer of the scene into consideration, there is another coordinate space: the view coordinates. In Fig.

2 the viewer is symbolized by the camera. The Z axis of the view coordinates always points directly to the viewer in orthographic projection. The X axis points to the right, the Y axis points upwards (**Fig. 3**).

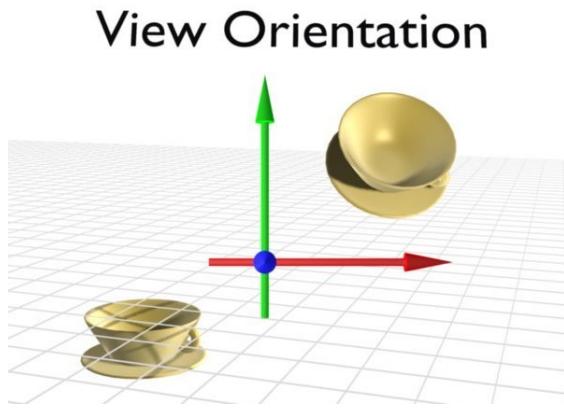


Figure 3: View coordinates in viewing direction

In fact you always work in view coordinates if you don't set it any other way*. This is particularly useful if you have aligned your view prior to modeling something, e.g. if an object has a slanted roof and you want to create a window to fit in that roof, it would be very complicated to build the window aligned to the local coordinate system of the object, but if you first align your view to the slanted roof, you can easily work in that view coordinate system.

(* In the Blender 2.6 series, the default has been changed to global coordinates. View coordinates remain as an option.)

If you work in one of the three standard views (Front/Top/Side) the alignment of the view coordinates fits the global coordinates. Therefore, it is quite natural to model in one of the standard views and many people find this the best way to model.

1.7.3 Normal coordinates

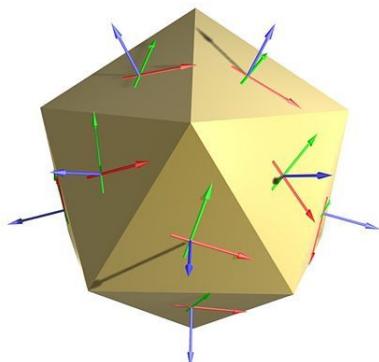


Figure 4: Normal coordinate spaces for faces. The normal is shown in blue.

Although Blender is a 3D program, only an objects' faces are visible. The orientation of the faces is important for many reasons. For example, in our daily lives it seems quite obvious that a book lies flat on a table. This requires the surface of the table and that of the book to be parallel to each other. If we put a book on a table in a 3D program, there is no mechanism that forces these surfaces to be parallel. The artist needs to ensure that.

The orientation of a face can be described with the help of the so-called surface normal. It is always perpendicular to the surface. If several faces are selected, the resulting normal is averaged from the normals of every single face. In **Fig. 4** the normal coordinates of the visible faces are drawn.

This concept can be applied to individual points on the object, even if the points themselves have no orientation. The normal of a point is the average of normals of the adjacent faces.

The images for this tutorial were produced with Blender v2.46.

1.7.4 UV Coordinates

In later parts (for example, talking about textures) you will come across coordinates labelled "U" and "V". These are simply different letters chosen to avoid confusion over "X", "Y" and "Z". For example, a raster image is normally laid out on a flat, two-dimensional plane. Each point on the image can be identified by X and Y coordinates. But Blender can take this image and wrap it around the surface of a 3D object as a texture. Points on/in the object have X, Y and Z coordinates. So to avoid confusion, the points on the image are identified using U and V to label their coordinates instead of X and Y. We then refer to "UV mapping" as the process of determining where each (U, V) image point ends up on the (X, Y, Z) object.

1.8 Overview

Blender's **user interface** (the means by which you control the software) is not particularly easy to learn. However, it has improved over time and is expected to continue doing so. The current version of the Blender software is 2.79. You can [download it](#) from the Blender Foundation's website.

The tutorials in this section will familiarize you with the basics of the user interface. By the end of this section, you should be able to:

- resize, split, and merge any Blender window;
- change the type of any Blender window;
- access user preferences;

- access panels containing buttons and other controls;
- change the viewpoint of a viewport.

For those new to Blender, this is a fundamental section of the book.

Advice on Customization

Blender is a complex software package with many customizable features. You can customize the user interface to assign new functions to buttons and hotkeys. In fact, you can change almost anything to suit yourself. However, this complicates the giving and following of directions. It is recommended you adhere to the default screen arrangements of Blender in order to be able to follow the remaining parts of these tutorials. Blender ships with 4 to 5 screen-content arrangements which are suitable for almost any kind of job you'll want to use it for - from creating motion and animation to making games.

We recommend leaving Blender's user interface in its "factory settings" while working through the *Noob to Pro* tutorials. At the very least, wait until you've mastered the basics before you customize the interface - and we know you definitely will when you master it!

1.9 Keystroke, Button, and Menu Notation

As you read through these tutorials, you will encounter cryptic codes such as SHIFT + LMB and *Timeline* → *End Frame*. They describe actions you perform using the keyboard and mouse. The notation used in this book comes from the standard used by the Blender community. We will try to import those standards here to facilitate our studies.

If you're reading this book online, you may wish to print this page for future reference. In addition, you can bookmark it in your browser for faster reference.

1.9.1 Hotkeys

Most computer keyboards have number keys in two different places. A row above the letters, and in a numpad (numeric keypad) to the right of the keyboard. While many applications use these two sets of keys interchangeably, Blender does not. It assigns different functions to each set. If you're using a laptop keyboard without a separate numeric keypad, this might cause some difficulty. You'll need to use your *function key* to do some things. It is possible to indicate to Blender the type of keyboard you are using, but we strongly recommend you use a standard external keyboard if you use a laptop for these tutorials



A typical numpad

as it will make your studies and usage of Blender much more straightforward and enjoyable.

This book often assumes your keyboard has a numpad. If yours doesn't, consult the tutorial on [Non-standard Input Devices](#) for alternative ways to access the numpad's functions.

Notation

Combinations that involve holding down a key while performing another action are written with a plus sign (+). Thus:

- Shift + Tab means to Tab while holding down Shift and
- Shift + Ctrl + F9 means to F9 while holding down both Ctrl and Shift .

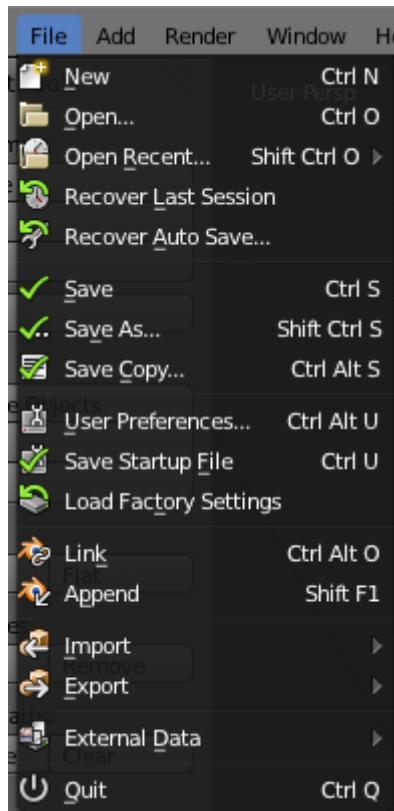
1.9.2 Mouse Notation

Blender uses three mouse buttons and the scroll wheel, if you have one. If your mouse only has one or two buttons, consult the tutorial on [Non-standard Input Devices](#) for alternative ways to access the functions assigned to these buttons.

Mouse and keyboard actions are often combined. Shift + RMB means to click RMB while holding down Shift .

1.9.3 Navigating Menus

Blender uses both pop-up and pull-down/pull-up menus. Many menus have **sub menus** (menus that are reached via another menu). If a menu item displays a triangle, that means it leads to a sub menu. For instance, the pull-down menu shown below has seventeen items, four of which lead to sub menus:



You can move through items in a menu by:

- moving the mouse pointer up and down

or

- pressing Up Arrow and Down Arrow

You can enter a sub menu by:

- moving the mouse pointer to the right

or

- pressing Right Arrow

You can leave a sub menu by:

- moving the mouse pointer to the left

or

- pressing Left Arrow

To initiate a menu action, you can:

- click LMB

or

- press Enter

You can escape from a menu by:

- moving the mouse pointer away from the menu

or

- pressing Esc

For each menu, Blender remembers your last choice and highlights it for you the next time you enter the menu.

Notation

Menu notation is fairly self-explanatory.

Shift + A Mesh → UV Sphere

Means:

1. Press Shift+A
2. In the menu that pops up, move through the items until *Mesh* is highlighted
3. Enter the *Mesh* sub menu
4. Move through the items until *UV Sphere* is highlighted
5. Press Enter or click the left mouse button to initiate the action

1.9.4 Additional Resources

- the [Blender Manual](http://wiki.blender.org/index.php/Doc:Manual/Interface/Keyboard_and_Mouse) page on “keyboard and mouse” at http://wiki.blender.org/index.php/Doc:Manual/Interface/Keyboard_and_Mouse

1.10 Non-standard Input Devices

This module is applicable only to users with non-standard input devices. If you have both:

- a three-button mouse

and

- a keyboard with a numpad,

you can skip this module.

Keyboards lacking a numpad

Most modern laptops have a pseudo-numpad, a set of keys in the main keypad which double as a numpad. The keys typically used for this purpose are:

When used as a pseudo-numpad, these keys typically act as the following keys from a true numpad:

The numpad functions of these keys can often be toggled with F11 or NUMLOCK on PCs or with F6 on Macs. Alternatively, you can often temporarily activate the numpad behavior by holding down Fn .

If your keyboard has the alternate labellings but you don't know how they work, consult your laptop owner's manual.

As a last resort, you can use the "Emulate Numpad" feature of Blender. This will allow you to use the normal numeric keys as if they were numpad numerics. Instructions for enabling this feature may be found in the "User Preferences Windows" module.

Blender uses the numeric keypad quite a bit. If you envision using your laptop for this kind of work, it may be worth investing in a *USB Numeric Keypad*. On eBay, prices for simple external numpads start around \$10 USD.

Non three-button mouse

For single-button mouse users, make sure that *View & Controls (Input* for Blender 2.54) (under "User Preferences" on the left-most drop-down menu) → *Emulate 3 Button Mouse* is enabled.

On many computers with two-button mice, MMB can be emulated by simultaneously clicking LMB and RMB . On Windows machines you'll need to enable this in the mouse settings in the Control Panel. On a Mac, open the *Keyboard and Mouse* preference pane and enable *Use two fingers to scroll*. Alternatively, by selecting *Emulate 3 Button Mouse* under *User Preferences*, MMB can be emulated by simultaneously clicking Alt and LMB .

Recent IBM Thinkpad laptops allow you to disable the 'UltraNav' features of the middle mouse button in order to use it as a 'normal' third button. Alternatively, some laptops allow areas (called gestures) on the movement pad to act as MMB or RMB , and these can be set up in the Control Panel in the Mouse Pointer options, selecting gestures and editing features there.

Apple single-button mouse While Mac OS X natively uses both the Ctrl + MB and Cmd + MB to emulate RMB , recent Blender releases for Mac OS X use only Cmd + MB for this purpose. This behavior is documented in the *OSX Tips* file that comes with the Mac version. You can also set the mouse to sense a right-click in System Preferences.

Note also that in the new, "unibody" design, the mouse

button is under the trackpad, and the shortcut for RMB is clicking with two fingers simultaneously, which can be enabled in the System Preferences.

Laptops lacking a middle button but with a smart-pad

Many laptops have smart pads. Smart-pads can use gestures to give the effect of MMB . The default for an Elan® Smart-Pad is two-finger tapping equivalent to clicking a MMB . Dragging two fingers is the same as turning a mouse wheel.

Tablet PCs

To get the effect of MMB in a viewport, drag your pen around while holding down the Alt key.

Additional Resources

- version 2.4: keyboard and mouse (old documentation) — Blender 2.4 Manual
- Input devices (version 2.7x)

1.11 Operating System-specific Issues

This tutorial covers user-interface issues that are specific to particular operating systems or window managers. Read the section that applies to your computer; you may skip the rest.

1.11.1 GNU/Linux

Alt + LMB is used for changing the angular view on two angular axes of the 3D View window, if Alt + LMB moves the current window, then there's a conflict with your window manager. You can resolve the conflict or use Ctrl + Alt + LMB or MMB instead. (Also, you may have activated Compiz->Rotate Cube. Default configuration for rotating the Cube is also Ctrl + Alt + LMB ; you may have to change this binding to an alternative configuration.) If you are running KDE this can be resolved by: RMB on the title bar of the main Blender window → select *Configure Window Behavior* → go to *Actions* → *Window Actions* → in the *Inner Window, Titlebar and Frame* section → select the Modifier key to be Alt and set all the select boxes beneath it to *Nothing*. An alternate method within KDE might be to RMB click on the title bar of the main Blender window; then select *Advanced* → *Special Application Settings...* → *Workarounds* and then click *Block global shortcuts* with *Force* selected and checked.

In Gnome, Click *System* → *Preferences* → *Window Preferences*. Look for the last three options *Control*, *Alt* and

Super. Select *Super*. Now you can press and hold Cmd to drag windows around, and use Ctrl and Alt as normal.

KDE

Under KDE, Ctrl + F1 through Ctrl + F4 are by default configured to switch to the corresponding one of the first four desktops, while CTRL + F12 brings up Plasma settings. You can change these in System Settings.

Gnome

You'll want to disable the *Find Pointer* functionality in Gnome, which will impair your ability to use certain functions such as *Snap to grid* and the lasso tool. If your mouse pointer is being highlighted when you press and release Ctrl , go to: *Mouse* in Gnome's *Desktop Settings* and uncheck the box *Find Pointer*.

Ubuntu

As of Ubuntu versions prior to about 09.10 (“Karmic Koala”), there was a known incompatibility between Blender and the Compiz Fusion accelerated (OpenGL) window manager used in Ubuntu. By default, Compiz Fusion is enabled in Ubuntu, causing the problems to manifest themselves in Blender as flickering windows, completely disappearing windows, inconsistent window refreshes, and/or an inability to start Blender in windowed mode.

The fix for this is simple. Install compiz-switch (might be in universe). Go to *Applications → Accessories → Compiz-Switch*. This will disable compiz temporarily. Do the same to turn compiz back on when you're done using Blender.

This is no longer needed for current releases of Ubuntu.

1.11.2 Mac OS X

On Macs with the new thin Apple Keyboard, you may need to press Fn in order to use the F1 through F12 keys.

To expand a section in Blender, you would usually press Ctrl + UpArrow . On a Mac, if “Spaces” is enabled, you may have to use Ctrl + Alt + UpArrow .

NOTE: On Yosemite 10.10.1 with “Spaces” enabled and Blender 2.72b, Ctrl + UpArrow doesn't seem to work.

1.11.3 Microsoft Windows

Two Ways to Launch Blender

Blender requires a console for displaying error messages, so if you launch Blender by means of an icon,

two windows will appear: the graphical user interface plus a console window. Closing either window will terminate Blender. These windows are indistinguishable in the Windows taskbar in versions of Windows before Windows 7, which leads to confusion. Also, launching this way does not provide any way to pass command-line arguments to Blender.

Launching Blender from a command prompt is extra work, but it overcomes these issues:

1. *Start → Run...*
2. enter cmd
3. enter cd c:\Program Files\Blender Foundation\Blender
4. enter blender

EDIT: Blender version 2.6 onwards doesn't have this problem, and hides the console window by default. You can show it by clicking Window > Toggle system console

Sticky Keys

Pressing Shift five times in a row may activate StickyKeys, an accessibility option which alters how the computer recognizes commands. If a StickyKeys dialog box appears, you should LMB the “Cancel” button.

If you don't need the accessibility features, you can disable sticky keys:

1. *Start → Control Panel*
2. double-click on *Accessibility Options*
3. LMB the *Keyboard* tab
4. for each of the options *StickyKeys*, *FilterKeys*, and *ToggleKeys*:
 - (a) clear the *Use ...* checkbox
 - (b) LMB the *Settings* button
 - (c) uncheck the *Use Shortcut* checkbox in the settings
 - (d) LMB the *OK* button for the settings
5. LMB the *OK* button for *Accessibility Options*

Multiple Keyboard Layouts

On systems with multiple keyboard layouts, pressing Shift + Alt can alter the layout. (For instance, it might change from QWERTY to AZERTY or vice versa.) Because of this issue, *Noob to Pro* avoids Shift + Alt hotkeys.

If you find your keyboard layout altered, press Shift + Alt again to change it back.

You can also disable the hotkey:

1. Start → Control Panel

2. double-click on *Regional and Language Options*

3. LMB the *Languages* tab

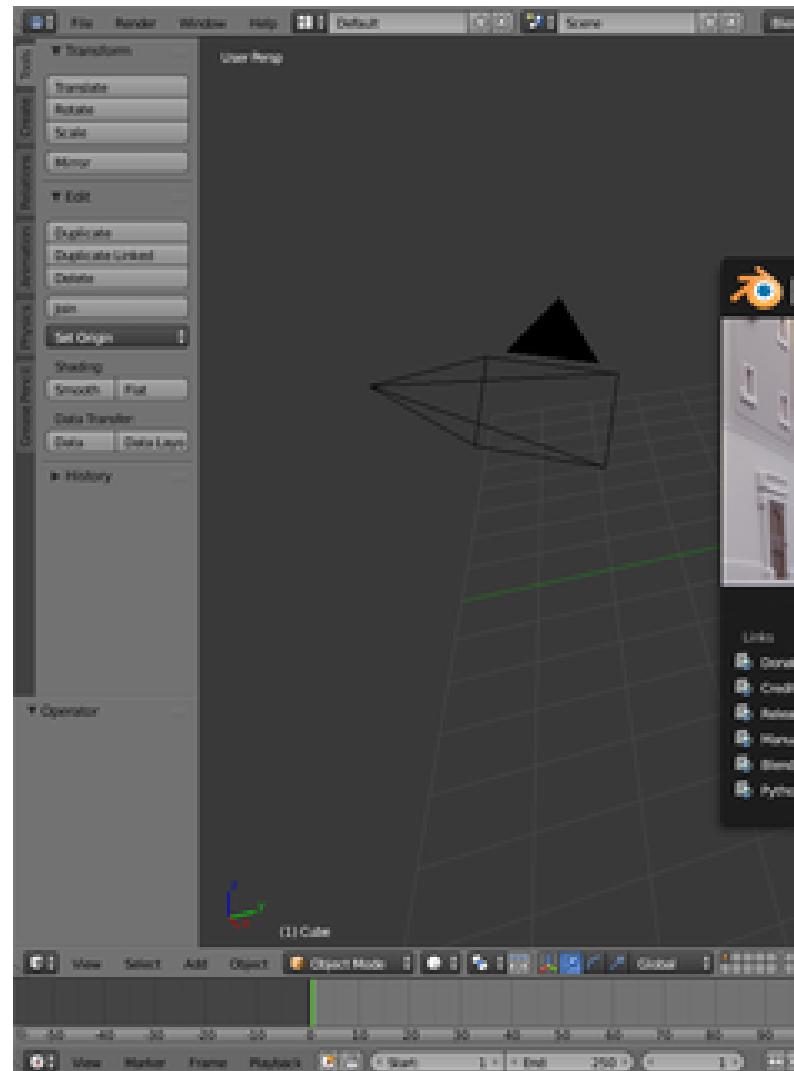
4. LMB the *Details* button

5. LMB the *Key Settings* button

6. LMB the *Change Key Sequence* button

7. uncheck the *Switch Keyboard Layout* checkbox

8. LMB the *OK* button



Additional Resources

- Input method editor keyboard shortcut (CTRL+SHIFT+0) switches the input language in Vista — Microsoft Support Knowledge-Base
-  StickyKeys at Wikipedia.

1.12 Blender User Interface

Here's a preview screenshot of Blender's interface.

For those familiar with older (pre-2.5x) versions of Blender, this will look very different. The redesign makes it much easier to find things.

For a detailed rationale explaining the redesign, read [this](#).

1.12.1 Why Doesn't It Follow UI Conventions For [Insert OS Of Choice Here]?

Blender follows its own user interface conventions. Instead of making use of multiple windows as defined by your particular OS/GUI, it creates its own “windows” within a single OS/GUI window, which is best sized to fill your screen. Many people accustomed to how applications normally work on their platform of choice, get annoyed by Blender's insistence on being different. However, there is a good reason for it.

The essence of the Blender UI can be summed up in one word: *workflow*. Blender was originally created by a 3D graphics shop for their own in-house use. Being a key revenue engine for them, they designed it for maximum productivity, speed and smoothness of operation. That means avoiding “bumps” that slow down the user. For example, windows never overlap, so there's no need to keep reordering them. You don't have to click in a window to make it active, just move the mouse. There is a minimum of interruption from popups asking for more information

before performing some action. Instead, the action is immediately performed with default settings, which you can adjust afterwards and get immediate feedback on the results.

Blender may not be “intuitive” to start learning, in that you cannot simply sit down in front of it and figure out things on your own, especially from a position of knowing nothing at all. But once you have picked up some basic conventions, you will find it starts to make sense and *then* you will be free to experiment and discover things on your own.

1.12.2 Why Doesn't It Prompt To Save Changes?

Most modern applications will ask for confirmation if you try to close a document that has unsaved changes. Blender is frequently criticized for not doing so.

But think about it. What constitutes an “unsaved change”? Are switching tools or adjusting the window layout changes worth saving? In Blender’s case, the answer is “yes”, because all that is part of the document state to be restored upon opening. So Blender would have to prompt for confirmation practically *every time* you closed a document or quit Blender.

Instead, Blender always saves changes when it closes, to a file called ‘quit.blend’. The next time you use Blender, simply select *File --> Recover Last Session* and you can resume right where you left off.

1.13 Blender Windowing System

The Blender user interface may appear daunting at first, but don’t despair. This book explores the interface one step at a time.

In this module, you’ll learn about Blender windows:

- recognizing windows and their headers,
- the different types of windows,
- how to activate and resize windows,
- how to split and join windows.

You’ll also practice launching and leaving Blender.

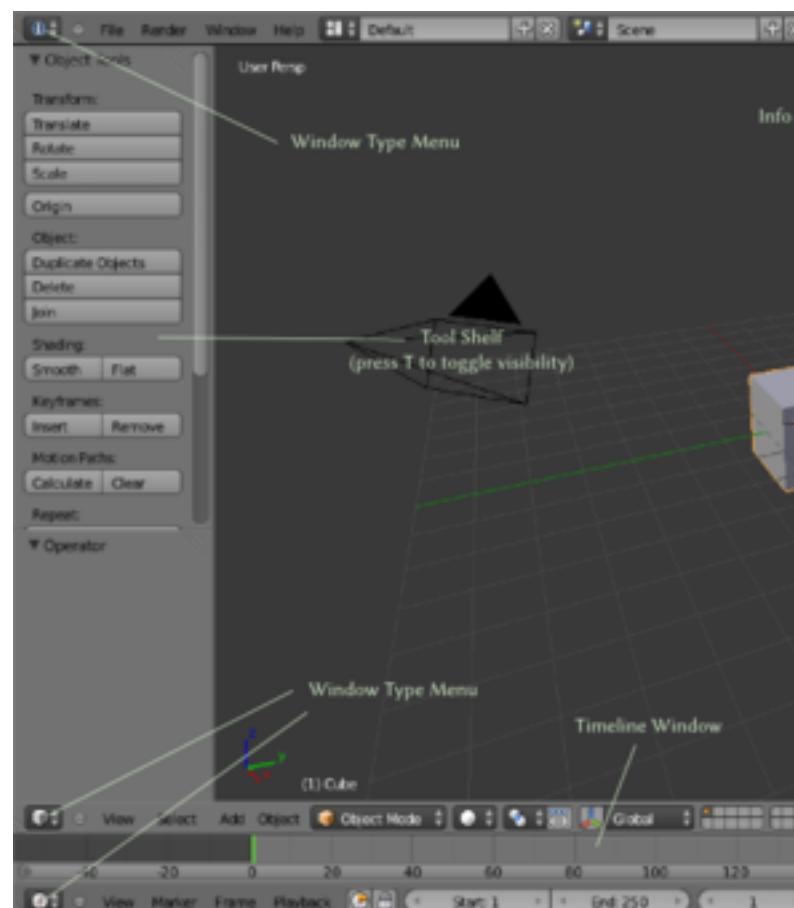
1.13.1 An Interface Divided

Blender’s user interface is divided into rectangular areas called windows (or sometimes, areas). The overall arrangement of windows is called a workspace.

If you haven’t already launched Blender, go ahead and do so. You should soon see something that resembles the following.

Blender has had some major changes to its user interface (UI) since version 2.4x. Some of these changes include moving buttons and changing the space bar hot key from the “add menu” to the “search menu” (SHIFT + A is now the “add menu” hot key). This is important to know when trying to follow tutorials.

Other changes include the addition of the tool bar and window splitting widget. The shelf widget (indicated by a plus sign) opens hidden tool shelves. The object tool shelf can be toggled on and off by pressing T . The properties tool shelf can be toggled on and off by pressing the N . The split window widget allows you to split and join windows. Blender 2.69 is shown below.



If you see something substantially different...

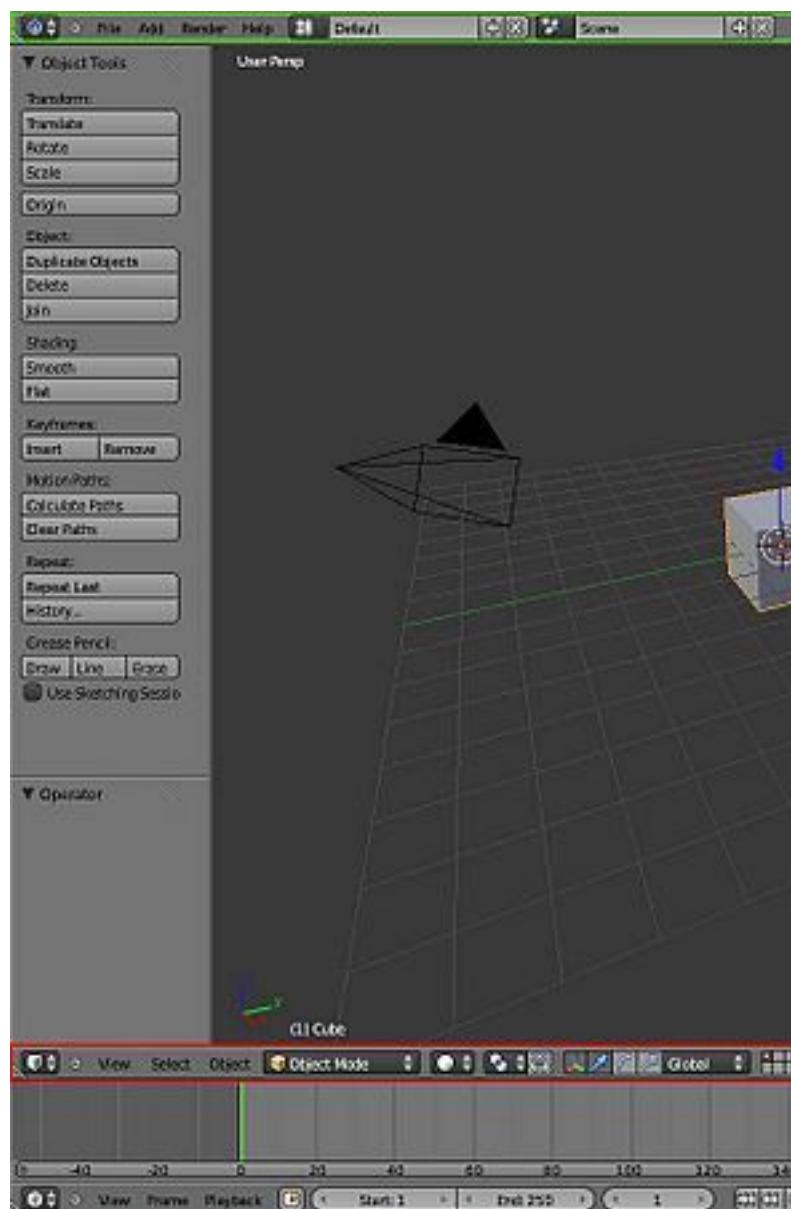
- You may be running a different version of Blender - perhaps a newer version. The screenshot was made using the 2.69 release.

If you’re running an older version, you should probably upgrade. Download instructions are in the Introduction.

- The user-interface settings on your computer may have been changed.

Try resetting the user interface with *File* → *Load Factory Settings*.

To take a screenshot in Blender, press Alt + F3 , and click *Make Screenshot*. This will record what's on your screen until you click the red *Close* button on the info header. The screencasts will be saved in the *tmp* folder. In Microsoft Windows, the *tmp* folder is located at 'C:\tmp'.



The header of the Info window is outlined in green.
The header of the 3D View window is outlined in red.
Note that it runs along the *bottom* of the 3D View window, not the top.

The header of the Properties window is outlined in blue.
The header of the Outliner window is outlined in white.
The header of the Timeline window is the one on the bottom (not outlined)

If you click with RMB on the header, a menu pops up which lets you move the header (to the top if it's at the bottom, or vice versa), or maximize the window to fill the entire workspace:

To hide the header completely, move the mouse to the edge of the header furthest from the edge of the window (i.e. the top edge of the header if it is at the bottom of the window, or vice versa); it will change into a vertical double-headed arrow. Now click with LMB and drag towards the window edge, and the header will disappear. In

1.13.2 Window Headers

Did you find all five headers?

Every Blender window has a header. A header can appear at the top of the window, at the bottom of the window, or it can be hidden. Let's take a closer look at the headers.



its place, you will see the following symbol appear at the corner of the window: . Click this with LMB to bring the header back.

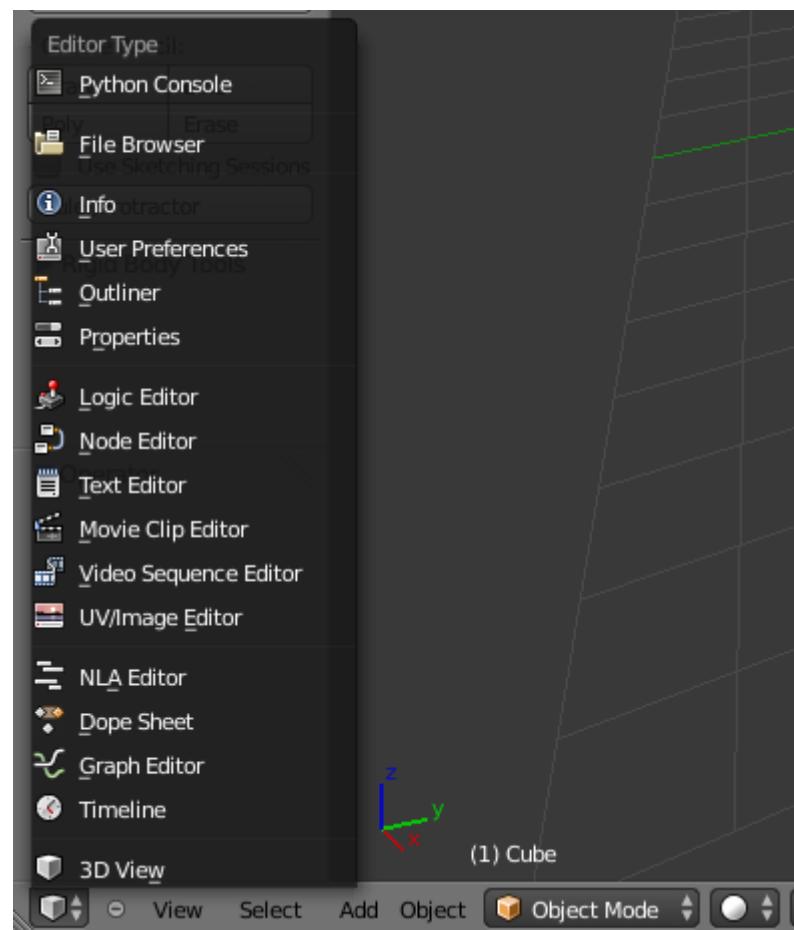
1.13.3 Window Types

Blender has many types of windows (there are 16 of them in Blender 2.69) and a Console for the Python programming language. You've just encountered the Info, 3D View, Properties, and Outliner windows. The rest will be introduced as needed in later modules.

Every window header in Blender has an icon at the left end to indicate the window type. For instance:

- = Info
- = User Preferences
- = 3D View
- = Outliner
- = Properties

If you LMB on the icon, a menu will pop up. (If you don't know what LMB means, please review the [Keystrokes, Buttons, and Menus Notation](#) module.)



By matching the icon in the header to the icons in the menu, you can tell that the window here is a 3D View window.

The menu can be used to alter a window's type. In this screenshot, the user is about to change the window into a Properties window.

If you've changed any window's type, please change it back (or reload the factory settings with *File → Load Factory Settings*) before continuing with this tutorial.

1.13.4 The Active Window

The **active window** is the one that will respond if you press a key. Only one Blender window is active at any given time.

The active window is usually the one containing the mouse pointer. (Blender uses a “focus follows mouse” user interface model. When a hotkey fails to work as expected, it is often because the mouse pointer has strayed into a neighboring window.) To change the active window, simply move the mouse pointer into the window you wish to activate.

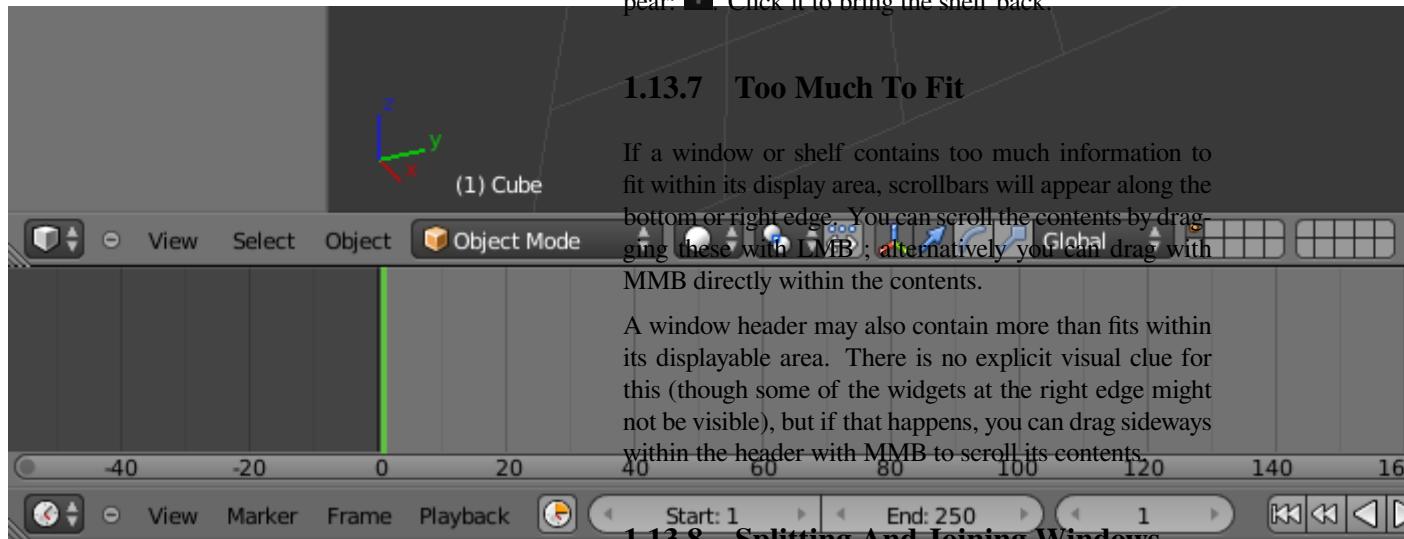
Practice changing the active window by moving your mouse between the 3D View and the Timeline windows. The Timeline window is directly below the 3D View header. At this point, it's worth mentioning that the

header for the 3D View window and Timeline window is at the BOTTOM of its own window instead of the top as the name “header” implies.

1.13.5 Resizing Windows

Resizing windows is easy.

Dragging on a Border



Step 1

Move the mouse pointer to the border between two windows (the area outlined in red below). The pointer will change to an up/down arrow.

Steps 2-4

Press and hold LMB .

Drag with the mouse to move the border up and down.

When the border is where you want it, release LMB .

Whenever you increase the size of one window, you decrease the size of another. That's because Blender has a non-overlapping window interface: unlike many other programs, it does not permit windows to overlap.

Maximizing a Window

Another way to resize a window is to maximize it. When Blender maximizes a window, it makes the window as large as possible. The previous window configuration is saved.

- To maximize the active window, press **Ctrl + UpArrow** , **Ctrl + DownArrow** or **Shift + Space** .
- When a window is maximized, use **Ctrl + UpArrow** , **Ctrl + DownArrow** or **Shift + Space** to restore the previous (unmaximized) window configuration.

Practice maximizing and un-maximizing the 3D View and Timeline windows.

1.13.6 Shelves

You will notice that the 3D View window (the largest window in the screenshots above) has several buttons down the left side. This rectangular portion is called the *Tool Shelf*. This is like a window within a window - you can drag the boundary between it and the main part of the 3D View to resize.

If you drag all the way to the window boundary, the shelf will disappear. In its place, the following symbol will appear: Click it to bring the shelf back.

1.13.7 Too Much To Fit

If a window or shelf contains too much information to fit within its display area, scrollbars will appear along the bottom or right edge. You can scroll the contents by dragging these with LMB ; alternatively you can drag with MMB directly within the contents.

A window header may also contain more than fits within its displayable area. There is no explicit visual clue for this (though some of the widgets at the right edge might not be visible), but if that happens, you can drag sideways within the header with MMB to scroll its contents.

1.13.8 Splitting And Joining Windows

At the top right and bottom left of every window, you will see something like this: If you move the mouse over the icon, you will see the pointer turn into a cross. At that point, you can do one of the following by clicking and dragging with LMB :

- Split the window into two copies horizontally by dragging horizontally away from the edge.
- Split the window into two copies vertically by dragging vertically away from the edge.
- Join the window to the adjacent one horizontally (getting rid of it and taking over its space) by dragging towards it.
- Join the window to the adjacent one vertically (getting rid of it and taking over its space) by dragging towards it.

Of course, the last two are only possible if there *is* in fact another window in that direction. Note: you can only join windows horizontally that are the same height, and windows vertically that are the same width.

1.13.9 The Default Workspace

If you look at the above screenshot of the default workspace, you will see the following window types:

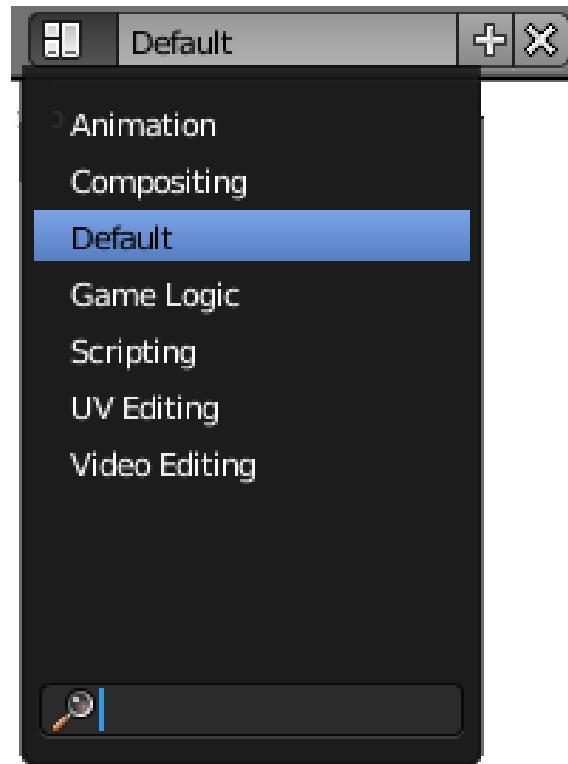
- The menu bar at the top (outlined in green) is actually a window, called Info . In previous versions of Blender, you could resize this to reveal the User Preferences, but in 2.5x they have been moved to their own window type. Instead, all you can see here if you enlarge the window are some debug messages, which may be removed in a future version of Blender. As of 2.70, the debug messages are still present in this menu.
- The largest window on the screen is the 3D View . This is where you work on your model.
- The Properties  window is the tall area on the right; this is where most of the functions are located for performing operations on models, materials etc. In previous versions of Blender this was called the Buttons window. Over time, it evolved into a disorganized area that made it difficult to find things. It has been cleaned up significantly in 2.5x. Note that it defaults to a vertical layout, rather than the horizontal one of previous versions. The new design prefers a vertical layout, which better suits today's widescreen monitors.
- The Outliner  (at the top right) gives you an overview of the objects in your document. As your models get more complex, you will start to appreciate the ability to quickly find things here.
- The Timeline  (across the bottom) becomes important when you're doing animation.

The default layout may not be optimal. For example, if you're doing a static model or scene, not an animation, you can get rid of the Timeline. If you're doing heavy script development, you'll probably want the Console available to try things out. And so on.

1.13.10 Workspace Presets

In the Info window/titlebar, you will see a menu with an icon like this . Clicking on it with LMB will show the following menu:

Selecting from this menu lets you quickly switch between various predefined workspace layouts, tailored to various workflows. Try it and see. You can return to the default layout by selecting "Default" (but note that any changes you make to the layout are immediately associated with the name being displayed here). The menu has a search box at the bottom. Typing text here will restrict the menu to showing items containing only that text. It might not appear to have much use, but in a complicated project that needs dozens of different layouts, the search function could become very useful indeed!



The name of the currently selected item appears to the right of the menu icon. In the illustration above, this is "Default". Blender allows you to rename the current menu item by clicking on it with the LMB and typing a new name, so take care not to do so unless you actually want to rename the menu item. For example, if you replace the name "Default" with "MyDefaults", you will subsequently see that "MyDefaults" appears in the list of menu items.

Note also the "+" and "X" icons to the right of the menu; clicking "+" creates a new entry which is a duplicate of the last-selected entry, while clicking "X" gets rid of the currently-selected entry. You will see these conventions appear consistently in menus elsewhere in Blender's new, revamped interface.

1.13.11 One Document At A Time

Blender can only work with one open document at a time. To save changes to the current document, select one of the Save options from the File menu (or press Ctrl + S to save under the last-saved name). To open a new document (actually load a copy of your last-saved user preferences), select "New" from the File menu (or press Ctrl + N), and select "Reload Start-Up File" from the popup that appears, but be aware *this will not automatically save any changes to the previous document*.

1.13.12 Scenes

A *scene* is like a separate Blender-document within-a-document. Different scenes within the same document can easily share objects, materials etc. You can define them once and make different renderings and animations from them. You create, delete and switch scenes using the scene  menu in the info header. A new document starts by default with just one scene, called “Scene”.

1.13.13 Leaving Blender

To exit Blender:

1. If there's a tool active, press Esc to exit the tool.
2. Press Ctrl + Q . This brings up an *OK?* menu.
3. Confirm *Quit Blender* by clicking LMB or pressing Enter .

1.13.14 Additional Resources

- YouTube video on Splitting and Joining Windows in 2.49 at <http://www.youtube.com/watch?v=uYb1j8X-ulc>
- YouTube video on Splitting and Joining Windows in 2.59 at <http://www.youtube.com/watch?v=mGK1gwFhx9M>
- the Blender Manual page on “window types” at http://wiki.blender.org/index.php/Doc:Manual/Interface/Window_types
- the Blender Manual page on “changing window frames” at system/Arranging frames <http://wiki.blender.org/index.php/Doc:Manual/Interface/Window/system/Arranging> frames

1.14 User Preferences Windows

In this module, we'll take a closer look at the User Preferences window. In the process, you'll encounter three different user-interface controls: radio buttons, toggle buttons, and sliders.

1.14.1 Saving User Preferences

Most applications have a place to keep user-configured settings (including document defaults), separate from any user-created documents. Blender works in a slightly different fashion. All user-configured settings are saved in *every* document you create. Each time you create a

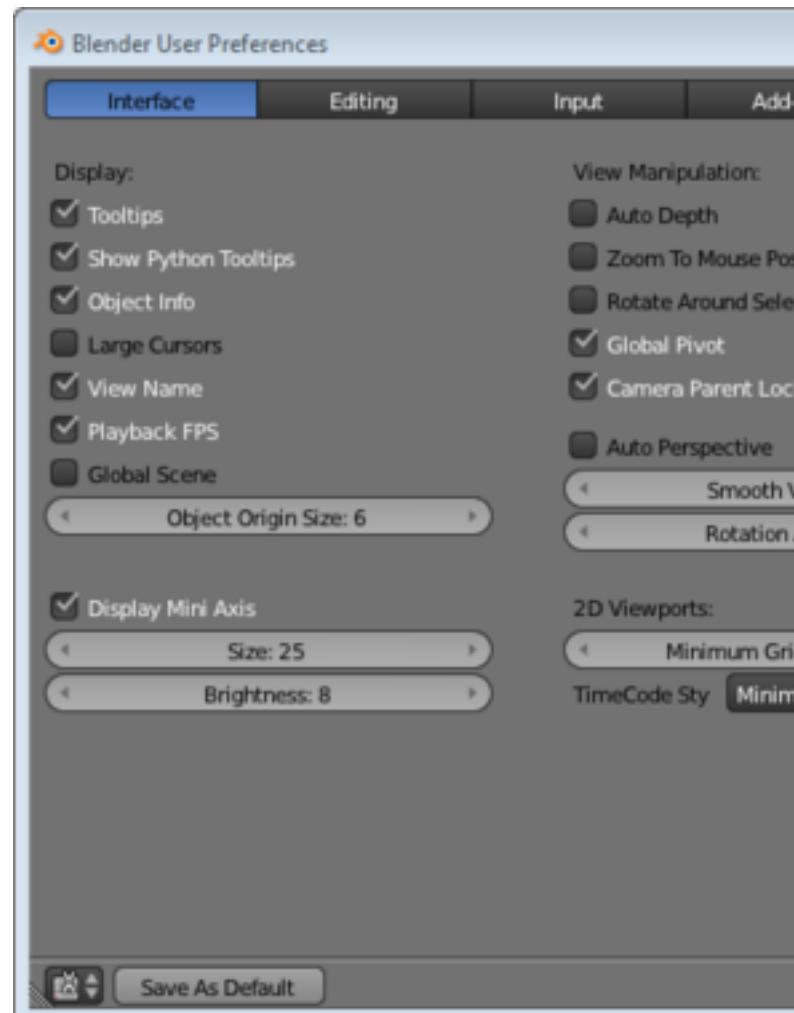
new Blender document, it reloads your *default* document, which is called *.B.blend*. To save your current state as the default document, press CTRL + U . This will save *everything* you've done to the current in-memory document, including objects and materials created, to *.B.blend*.

1.14.2 Accessing the User Preferences

First we must open the User Preferences window. There are 3 ways to do this:

- Click LMB *File → User Preferences...*
- Change the window type of the top header to *User Preferences*  and drag the header down.
- Press CTRL + ALT + U , which will open the User Preferences into a separate window which you can resize at will.

The User Preferences window should look something like the screenshot below.



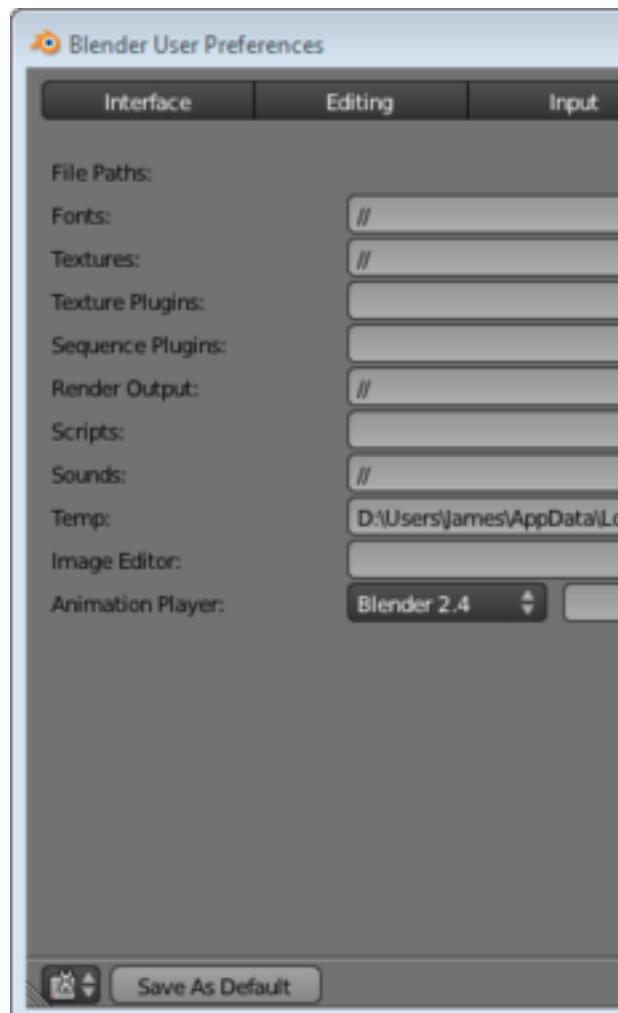
1.14.3 Configuring Your Preferences

In order to get to modeling and rendering sooner, this tutorial will cover only a few of the many user-settable preferences.

Auto Save

As the name suggests, Auto Save automatically saves the current .blend after a specified period of time. The settings are:

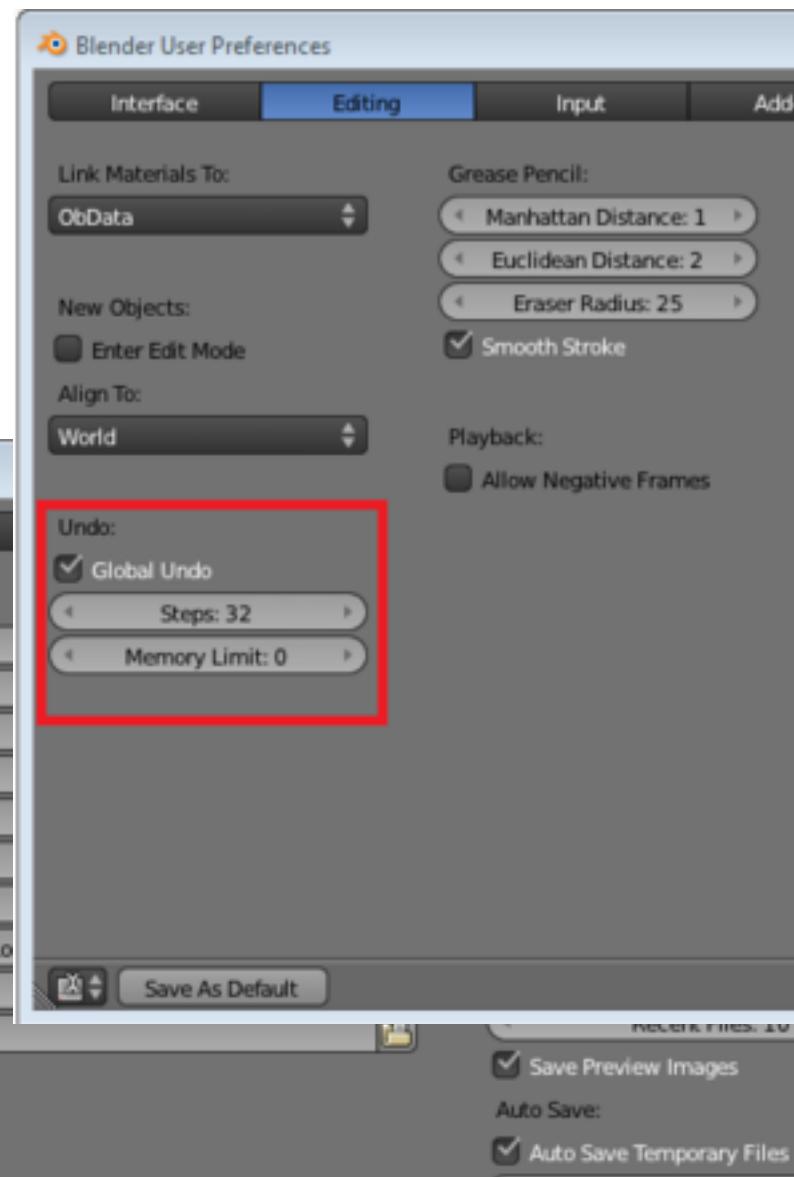
- *Auto Save Temporary Files*: This enables/disables the auto save feature.
- *Timer (mins) slider*: This specifies the time in minutes between each auto save.



Number of Undo Levels

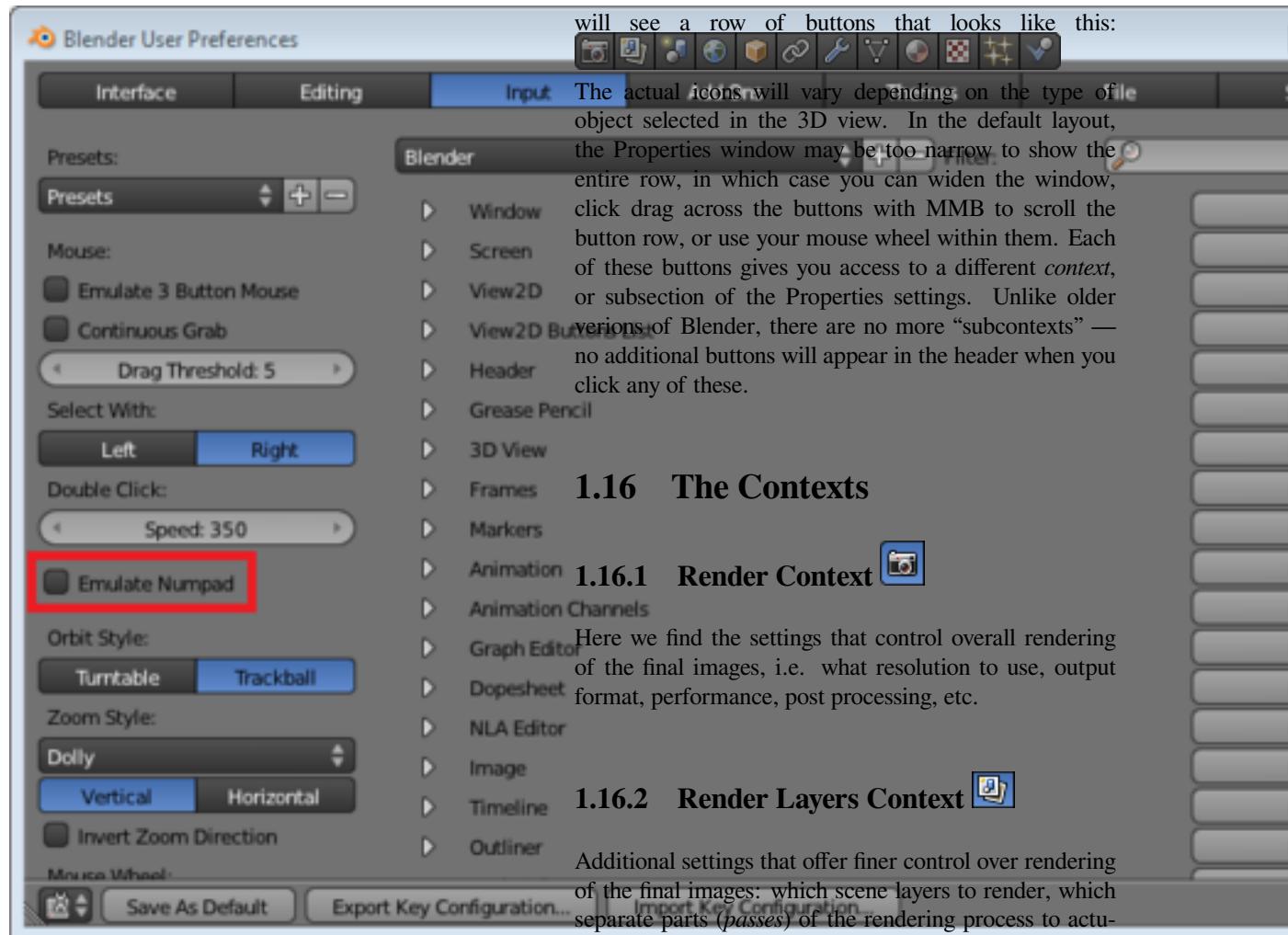
Next we'll look at the undo settings. By default, Blender remembers your last 32 actions and allows you to undo

them one at a time by pressing **Ctrl + Z**. If your computer has plenty of memory, you may wish to increase that number. If it has relatively little memory, you might consider decreasing it to 10 or 20. The *Memory Limit* slider specifies the amount of RAM (in megabytes) to use for storing the undo levels. Undo level "0" is unlimited.



Numpad Emulation

Blender uses numberpad keys (such as **7**) to control the 3D View and ordinary numeral keys (such as **7**) to change layers. If you are working on a laptop or if you find the numberpad inconvenient, you can select *Emulate Numpad* to reassign the 3D View controls to the ordinary numeral keys.



If you ever need to restore Blender to its factory settings, you can delete your personal ".blend" file then restart Blender, or click LMB *File → Load Factory Settings*

1.14.4 Additional Resources

- the Blender Manual page on “Configuration” at <http://wiki.blender.org/index.php/Doc:Manual/Interface/Configuration>
- The tutorial on Non-standard Equipment describes other workarounds for numpad issues.

1.15 Properties Window

The Properties  window is where you will find most of the functions that Blender can perform with objects and materials, animation, rendering, etc. It is the area where you will see the greatest number of changes from earlier versions of Blender (in which, it was called the Buttons window). Hopefully you'll agree the new layout makes it much easier and quicker to find things!

In the header of the Properties window, you

1.16 The Contexts

1.16.1 Render Context

Here we find the settings that control overall rendering of the final images, i.e. what resolution to use, output format, performance, post processing, etc.

1.16.2 Render Layers Context

Additional settings that offer finer control over rendering of the final images: which scene layers to render, which separate parts (*passes*) of the rendering process to actually perform, and how to group them into *render layers* (not to be confused with the scene layers) for input into subsequent compositing.

1.16.3 Scene Context

Contains settings for colour management, choosing which camera to use for rendering, and units and gravity settings for physical modeling.

You can also select another scene to be a “background” for this scene. That is, all renders of this (foreground) scene will also include the contents of the background scene, as though they had been copied into this scene. While the background appears in the 3D viewport when editing this scene, none of its contents are editable, or even selectable; that has to be done in the background scene itself.

1.16.4 World Context

Settings that govern the environment in which the model is rendered. e.g. background sky color, mist and star settings, lighting etc.

1.16.5 Object Context

Settings that apply to all types of objects. e.g. overall transformations, layer assignments, grouping etc. The settings shown here (and any changes made) apply to the last object selected. This is also the case for the following object-specific contexts.

1.16.6 Object Constraints Context

These settings limit the motion of the object for animation purposes. The limits can also be tied to the motion of other objects in various ways.

1.16.7 Object Modifiers Context

Settings for applying *modifiers* to the object geometry. These make changes to the geometry that only take effect at rendering time. Note: lamps, cameras and empty objects cannot have modifiers.

1.16.8 Object Data Context



Settings specific to the type of object, e.g. mesh vertex groupings, text font, lamp settings, camera settings, etc. This is reflected in the icon, which changes according to the type of object selected.

1.16.9 Material Context

The material settings for an object control its appearance, e.g. its colour, whether it has a shiny or dull surface, how transparent it is, and so on.

1.16.10 Texture Context

The texture settings specify patterns that break up the uniform appearance of a material. These patterns can affect the colour of the material, give it a rough surface, or modify it in other ways.

1.16.11 Particles Context

An object can be set to emit particles, like smoke, flames or sparks. The concept of “particles” (and the underlying algorithms) also includes the generation of hair or fur. Particles can be entirely custom objects, to produce effects like blades of grass interspersed with flowers in a

field, water droplets on a wet surface, or even scatterings of entire buildings to make up a large cityscape!

1.16.12 Physics Context

Settings that control how the object reacts to forces similar to objects in the real world, e.g. whether it behaves like a rigid body that keeps its shape but can be knocked around, something soft e.g. a pillow, or a flowing liquid.

1.16.13 Where Did The Old Stuff Go?

For those used to older (pre-2.5x) versions of Blender...

The old Logic Context has been moved into its own window type, the Logic Editor.

The old Script Context is gone. Python scripting is now much more closely integrated into the Blender UI, and old scripts will not work anyway.

The functions of the other contexts have been rearranged into the new contexts as listed above. Once you get used to the new arrangement, it should make much more sense than the old one.

1.17 3D View Windows

3D View  windows are used to visualize 3D scenes. You'll do a lot of work in these windows, so you will need to learn your way around.

In this module, you'll learn:

- to recognize 10 things commonly seen in viewports
- to tell which mode Blender is in
- how to change viewport options and viewpoints
- how to position the 3D cursor

You'll also learn the fundamentals of:

- visibility layers

1.17.1 The Viewport and its Contents

Aside from its header, the remainder of a 3D View window is its viewport. You use viewports any time you need an up-to-date view of the scene you're working on.

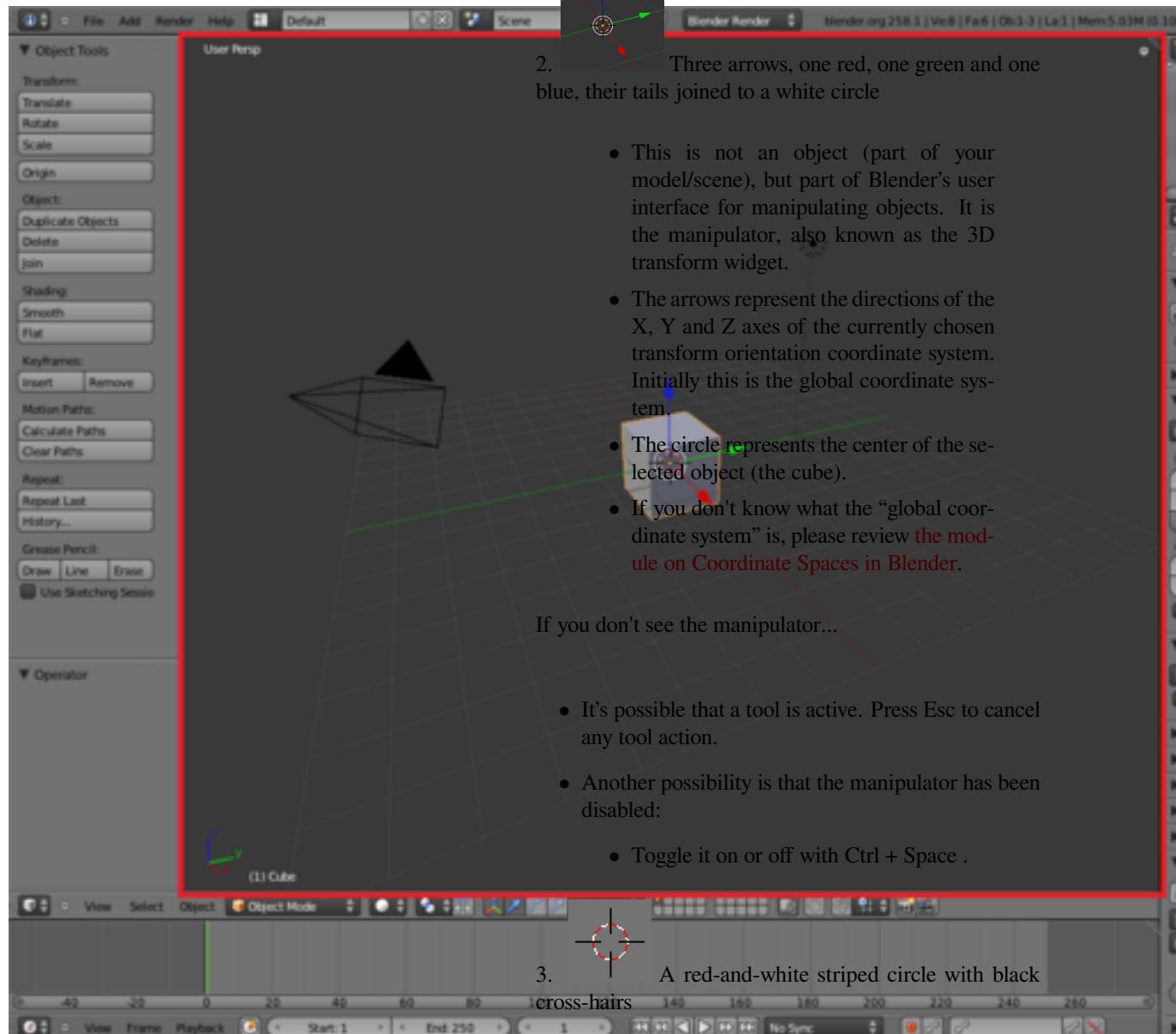
Viewports are busy places. Go on a scavenger hunt and see what you can find in a simple viewport.

1. Launch Blender.

2. Just so we're all looking at the same scene, load the factory settings using *File → Load Factory Settings*.
3. Confirm the “Load Factory Settings” popup with LMB (or Enter).
4. If the NumLock indicator on your keyboard is unlit, press NumLock so that numpad hotkeys will work properly.

(If you're unsure what LMB means, please review [the Keystroke, Button, and Menu Notation module](#).)

You should see something like this:

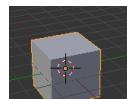


Here the viewport has been outlined in red to focus your attention on it.

A Virtual Scavenger Hunt

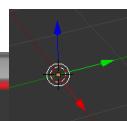
Look at the default scene and find the following eight items:

In the Center



1. a solid gray cube with orange edges.

- This is the default cube, your first Blender object!



2. Three arrows, one red, one green and one blue, their tails joined to a white circle

- This is not an object (part of your model/scene), but part of Blender's user interface for manipulating objects. It is the manipulator, also known as the 3D transform widget.
- The arrows represent the directions of the X, Y and Z axes of the currently chosen transform orientation coordinate system. Initially this is the global coordinate system.
- The circle represents the center of the selected object (the cube).
- If you don't know what the “global coordinate system” is, please review [the module on Coordinate Spaces in Blender](#).

If you don't see the manipulator...

- It's possible that a tool is active. Press Esc to cancel any tool action.
- Another possibility is that the manipulator has been disabled:
 - Toggle it on or off with Ctrl + Space .

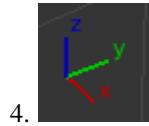


3. A red-and-white striped circle with black cross-hairs

- This is not an object. It is the **3D Cursor**, which indicates where newly-created objects will appear in the scene.

- The cursor is similar to the insertion point in a text editor, which indicates where new text will be inserted in a document.

In the Lower Left Corner



4.

- This is not an object. It is the **mini axis**, and its orientation matches that of the global coordinate system, with the usual conventions: red for X, green for Y and blue for Z. Think of it as a little compass, reminding you which way is left/right, front/back and up/down.

5. The notation "(1) Cube"

This is not an object. It is object info, indicating that:

- You're viewing the first frame of an animation.
- and
- The current or most recently selected object is named "Cube".

In the Upper Left Corner

6. The notation "User Persp"

This is not an object. This tells you which mode the viewport is in. The first word will change if you select one of the *perfect* views or the camera view (see below), otherwise it just says "User", and the second word is "Persp" or "Ortho" to indicate whether this is a perspective or orthographic view.

To the Right of Center

7.  A black round thing that resembles a sun symbol

This represents a **lamp**, a light source for the scene. (It is an object.)

8.  A pyramidal wireframe item

This represents a **camera**, a viewpoint that can be used for rendering. (It too, is an object.) The direction it is looking is out the base of the pyramid. The solid triangle attached to one side of the base is to remind you which way is up in the image that the camera takes.

On a small display, the camera might initially lie outside of the viewport and thus be invisible. In that case, zoom out by scrolling with MMB until it becomes visible.

Throughout

9. A dark gray background, divided into squares by lighter lines. This is the grid floor, which you can (but don't have to) use as a ground plane for positioning your models.

Each grid square is one **blender unit** (or **BU**) on a side. A BU can be whatever you wish, e.g. an inch, a centimeter, a mile, or a cubit. Blender lets you choose your scene scale in the Scene tab of the Properties Panel.

10. Three mutually perpendicular coloured lines associated with the grid floor: the red and green ones lying horizontally in the floor and the blue one running vertically. These are the global coordinate axes for orienting your scene. Red is the X-axis, green the Y-axis, and blue the Z-axis.

- In Blender 2.67a, you can't see the blue line for Z-axis here, but you **can** see it in Front or Side view.

1.17.2 Modes

Blender has many **modes**, i.e. settings that affect its behavior, and this is especially true of the 3D View window.

Sometimes it's not obvious which mode is active. This leads to **mode errors** where Blender will do something you didn't expect because you thought it was in one mode and it was actually in another.

The function performed by a hotkey or mouse button can depend on:

- what mode the user interface is in,
- whether the keyboard is in NumLock mode,
- which window is active,
- the mode the active window is in,

- which item or items are selected,
- whether you've initiated a hotkey sequence.

It helps to recognize the common modes and how to get out of them.

Object Mode vs. Edit Mode

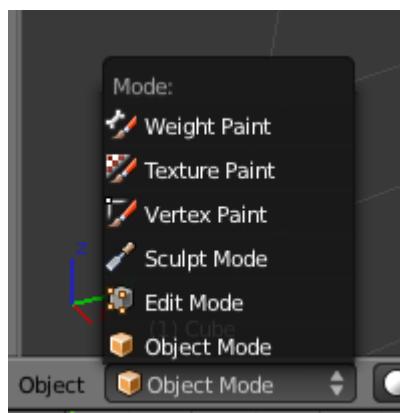
The 3D View windows are normally in Object Mode. In this mode:

- The mouse pointer is a white arrow (on macOS it is black).
- RMB is used to select objects in the scene

If there are objects in the scene, you can get into five other modes:

- Edit Mode: used to edit the shapes of objects
 - The mouse pointer is a thin inverse-video cross.
 - RMB is used to select vertices, faces or edges of the current object.
 - Press Tab to enter/exit this mode.
- Sculpt Mode
 - The mouse pointer is now a thin, orange circle.
- Vertex Paint
 - The mouse pointer is the same as in sculpt mode, a thin, orange circle.
- Texture Paint
 - The mouse pointer is a thin white circle.
- Weight Paint
 - The mouse pointer is again, a thin orange circle

These modes are also indicated by a menu in the 3D View header. You can use this menu to change modes.



These modes are a setting shared by all 3D View windows. In other words, when you change the mode in one window, any other 3D View windows change mode also.

1.17.3 Viewport Options

Solid vs. Wireframe

By default, the 3D View window draws objects using the Solid drawtype, in which surfaces are opaque. To toggle between Solid and Wireframe drawtype (edges only, no faces) for a particular viewport:

1. Activate the 3D View window.
2. Press Z .

Alternatively, you can choose these and other drawtypes from the “Viewport shading” menu in the 3D View window header.

Orthographic vs. Perspective

By default, viewports draw orthographic views. To toggle a viewport between orthographic and perspective views:

1. Activate the 3D View window.
2. Press Num5 .

(If you're unsure what the difference is, please review the “Orthographic Views” module and the “Perspective Views” module.)

Note this perspective versus orthographic setting for the 3D viewport is completely separate from the similar setting in the camera properties. The former takes effect while you're working on the model, the latter when you render.

So why have a separate setting for the 3D view? Because certain aspects of modelling are easier in one view than another. If the final render will be using perspective, then showing perspective in the 3D view naturally gives you a better idea of how the final render will look. But perspective foreshortening can sometimes make it hard to ensure the model has the proper shape, which is why there is the option to switch to orthographic view.

If you have trouble distinguishing between orthographic view and perspective view

... you should activate the View Name option. This is enabled by default and causes the name of the current view (“User Persp”, for instance) to appear in the upper left corner of every viewport. If there is no text, then you can enable it by:

1. Accessing the User Preferences window.

2. Click on the *Interface* tab.
3. Enable *View Name*.

1.17.4 Changing Your Viewpoint, Part One

Each viewport has a **viewpoint**, which takes into account:

- the location of the viewer in the 3D scene (There doesn't need to be an object at that location.)
- the direction the viewer is looking
- the magnification (or zoom factor) used

Changing your viewpoint allows you to navigate your way through a 3D scene.

We'll start with three very basic techniques:

- Zooming
- Orbiting/View Rotation
- Perfect Views.

Additional techniques will be covered later in this module.

Zooming

Blender offers several ways to zoom in and out:

- Use SCROLL
- Click and drag vertically with Ctrl + MMB .
- Use Num+ and NUM- to zoom in and out in small increments.

Note the following limitations of Blender's zoom feature:

- If the viewport is in orthographic mode, Blender zooms as if looking through a telescope. You can increase the magnification, but the viewpoint's location doesn't change. For this reason, you cannot zoom into or through objects in orthographic mode.
- If the viewport is in perspective mode, Blender zooms to the center of the viewport. The viewpoint can pass through objects, but can't pass beyond this point, no matter what you do. Zooming only gets slower and slower and slower. If the center of the viewport is somewhere you don't expect, zooming may appear to be broken.

Orbiting and View Rotation

Let's fly around in the default cube, viewing it from different angles. In this way you'll see that it really is a cube, centered on the origin, half above the X-Y plane and half below it.

1. Activate the 3D View window by placing the mouse pointer inside it.
2. Now you can:
 - Click and drag with MMB to orbit freely around the center of the view.
 - Use Shift + Alt + SCROLL to rotate the viewpoint vertically around the center of the view.
 - Use Num2 and Num8 to rotate the viewpoint vertically around the center of the view in 15-degree increments.
 - Use Ctrl + Alt + SCROLL to rotate the viewpoint around the Z axis.
 - Use Num4 and Num6 to rotate the viewpoint around the Z axis in 15-degree increments.

If this is all very confusing for you, don't worry! You'll learn as you get more experience.

When you are finished flying around the cube, you can restore the original view by reloading the factory settings with *File → Load Factory Settings*.

If the hotkeys don't work...

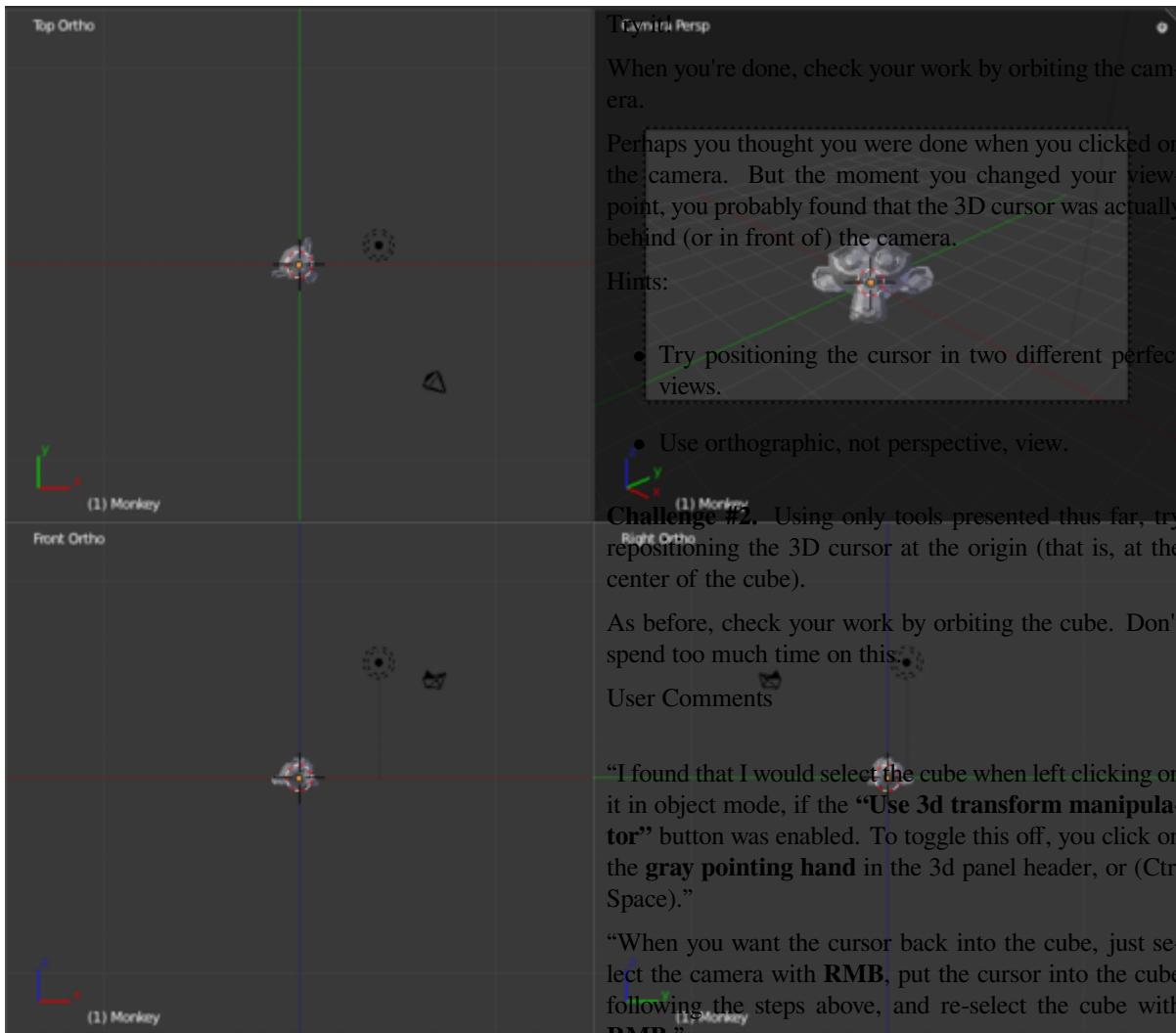
You may have pressed number keys above the letters instead of the ones on the numpad. If you do, the default cube will vanish. This is because the scene consists of multiple layers. The default cube is in layer 1, and you've told Blender to switch to the layer of the number you just pressed. The selected object (the cube in this case) remains in layer 1, which is no longer visible. For instance, 2Key tells Blender to switch to layer 2. To switch to layer 1 again, press 1Key . You can view the different layers by clicking on the little squares on the layer map:



Perfect Views

It's often useful to get a **perfect view** of a scene, i.e. to view it along one of the main axes, with the other two main axes oriented up-down and left-right.

The following screenshot shows all three perfect views plus camera perspective for the Suzanne primitive:



This layout is used so often, it has a keyboard shortcut: (**CTRL + ALT + Q**).

1.17.5 Positioning the 3D Cursor

Positioning the 3D cursor is a very basic operation, yet one that many beginners find challenging. It touches on an issue common to all 3D graphics software: “How do you specify points in a 3D scene when we can only see two dimensions at a time?”

Basic Technique

1. Go into either Object Mode or Edit Mode.
2. Move the mouse pointer to the desired position (in any viewport).
3. Click LMB .

Two Challenges

Challenge #1. Using only tools presented thus far, try positioning the 3D cursor on the virtual camera.

“To get it back in the cube: 1) Make sure you’re in object mode. 2) Select the cube. 3) Object > Snap > Cursor to selection (cursor refers to the 3D cursor here) so it puts it right in the middle of the cube.”

“I think it’s an essential point to note that *in order to place the cursor inside the cube, the cube must NOT be selected.* AKEY was probably the best way to deselect the object.”

“If I remember correctly, undo history gets cleared when you switch between object and edit mode.”

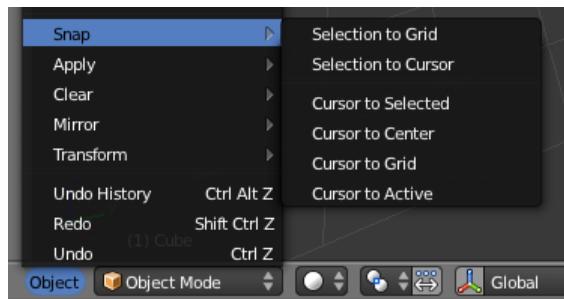
“I wasted a lot of time here. *Thank you* to the reader who suggested (on the 3D view header) Object > Snap > Cursor to selection. It was the only thing that worked to get the cursor visible again and placed where clicked.”

“I missed the point of the exercise first time around. You can’t set a 3D point on a 2D screen without technique. Orthographic views are crucial. I am just learning, but take that, at least, away from it.”

“Positioning the 3D cursor in orthographic views always made it snap to the cube surface, making it impossible to center precisely. Fix this by disabling “Cursor Depth” on the “interface” tab under “User Preferences”.

“The phrase *check your work by orbiting the camera* needs additional clarification, such as a referenced section or the precise commands to use.”

More Ways to Position the Cursor



Here’s an easy way to position the cursor at the center of an object:

1. Make sure Blender is in Object Mode, with the object selected.
2. Move the mouse pointer to any 3D View window.
3. Snap the cursor to the selected object using either:
 - Shift + S → *Cursor to Selected*
 - or
 - Object → Snap → *Cursor to Selected*

Here’s 2 easy ways to relocate the cursor to the scene’s origin (0, 0, 0):

1. Move the mouse pointer to any 3D View window.
2. Press Shift + C to reset the cursor to the origin.
 - Note that this also changes the view location, meaning that when you zoom in, you won’t zoom in to the scene origin.
3. A better way is to click *Object → Snap → Cursor to Center*
 - You can also do this by Shift + S → *Cursor to Center*.

1.17.6 Changing Your Viewpoint, Part Two

Now you’ll learn some additional techniques for obtaining the view you want:

- Panning
- Centering
- Jumping to the camera’s viewpoint
- Zooming in on a selected area

Panning

When you orbited the cube, the viewpoint’s position and direction both changed at the same time. You also can shift the viewpoint up-down or left-right *without* changing its direction. (This is similar to the side-scrolling effect in the classic Mario and Sonic video games.)

This is called *panning*, and it’s an important skill to master. Try it now:

1. Activate a 3D View window by placing the mouse pointer inside it.
2. Now you can:
 - Use Shift + SCROLL to pan up and down.
 - Use Ctrl + Num2 and Ctrl + Num8 to pan up and down in small increments.
 - Use Ctrl + SCROLL to pan left and right.
 - Use Ctrl + Num4 and Ctrl + Num6 to pan left and right in small increments.
 - Click and drag with Shift + MMB or Shift + Alt + LMB to pan freely in the viewplane.

You will likely find this to be a distraction in some cases. To move the viewpoint position back to the center, snap the cursor to the center, then click *View → Align View → Center View to Cursor*. You could

also snap the cursor to the center then press **Ctrl + Num .**

In version 2.74 use **Alt + Home.** to center the view to the cursor.

Centering

When you zoom or rotate the view, you always zoom or rotate around the center of the view.

To make sure everything in your scene is visible:

1. Press **Home .**

To center the view on an arbitrary point:

1. Move the 3D cursor to the point of interest.
2. Verify the cursor position from a second viewpoint.
3. Press **Alt + Home** to center the view.

To center the view on an object in the scene:

1. Make sure Blender is in Object Mode.
2. Zoom out until the object is in the viewport.
3. If any objects are selected, use **A** (or *Select → Select/Deselect All*) to deselect them.
4. Select the object of interest by clicking **RMB** on it.
5. Press **Num.** to center the view.

Jumping to the Camera's Viewpoint

To see the scene as the virtual camera sees it, press **Num0** . Afterwards, you can rotate, pan, and zoom normally, but the virtual camera will not follow. To go back to your previous view, press **Num0** again. (In the latest versions of Blender, the virtual camera can be made to follow all the changes made in viewpoint while in camera view by checking the option “Lock Camera to View” on the Transform panel. Hit **N** on your keyboard to bring up the transform panel. To disable this option uncheck “Lock Camera to View.”)

Zooming into a Selected Area

Suppose you want to get an extreme closeup of a particular area. Because there’s no center mark on the viewport, you might have to pan and zoom several times to get the desired view.

The shortcut for zooming to an area is:

1. Activate a 3D view window that contains the area of interest.

2. Press **Shift + B** . A crosshair appears in the viewport.
3. Click and drag with **LMB** to draw a rectangle around the area of interest.
4. When you release **LMB** , the viewport will zoom in on the area you selected.

1.17.7 View Navigation

You can also change your viewpoint in the 3D view by “walking” or “flying” through it. To activate this, press **SHIFT + F** . By default in Blender 2.70, this puts you in “walk” mode. Earlier versions only offered “fly” mode. (In Blender 2.70 and later, you can choose which one you prefer in User Preferences, under the Input tab.)

In both modes, helpful prompts appear in the header of the 3D view window to remind you of the key functions while the mode is in effect. When you have reached the position and orientation you want, press **LMB** or **ENTER** or **SPACE** to end the navigation mode and stay there, or **RMB** or **ESC** to abandon the navigation mode and be teleported immediately back to your original position and orientation.

Walk Mode

In this mode, you move the mouse to turn your view up/down/left/right, and **W , A , S** and **D** or the corresponding arrow keys to move forward, left, back or right, and **E** and **Q** to move up or down respectively. Hold a movement key down to keep moving. Movement stops as soon as you release it. Pressing **MMB** will “teleport” you close to whatever objects lie within the crosshairs at the centre of the view.

You can also use **TAB** to turn on gravity. Make sure there is a floor or other object under you to land on! With gravity on, you can no longer use the vertical movement keys, but you can use **V** to make jumps. Press **TAB** again to turn gravity off.

Fly Mode

In this older mode, moving the mouse to change the view works the same as in Walk mode, but the above direction keys (**W , A , S , D , E , Q** and the arrows) apply “thrust” in the respective directions, so you keep moving after releasing the key. Press the key repeatedly to increase your speed in that direction, or press the key for the *opposite* thrust direction to reduce your speed. You can roll the mouse wheel up to apply forward thrust, or roll it down to apply backward thrust.

Your current velocity vector automatically changes direction with you when you turn. Thus, you can apply a single burst of sideways thrust while facing an object, then,

without applying any additional thrust, keep turning to face the object, and you will go right around it.

1.17.8 Visibility Layers

Every object in the scene is assigned to one or more of 20 visibility layers.

Visibility layers have many uses:

- You can put scenery, characters, particles, and lamps in different layers, to help organize your scene.
- By changing which layers are visible, you can simplify your view of the scene and work with only one or two layers at a time.
- When rendering, only visible layers are included. You can use this to render your scene layer by layer, checking each layer separately.
- You can configure lamps to illuminate only objects in the same layer.



Left: Viewing layer 1 only.

Right: Viewing all 20 layers.

In Object Mode, you can tell which layers are visible by looking at the twenty small boxes located in the 3D View header between the Transform Orientation menu and the “Lock” button. The top row of boxes represents layers 1 through 10, with 1 being the leftmost and 10 being the rightmost. Similarly, the bottom row of boxes represents layers 11 through 20.

Hotkeys

- To view just one of layers 1 .. 9, press 1KEY .. 9KEY .
- To view just layer 10, press 0Key .
- To view just one of layers 11 .. 19, press ALT + 1KEY .. ALT + 9KEY
- To view just layer 20, press ALT + 0KEY .
- To toggle the visibility of one of layers 1 .. 9 without affecting the visibility of the other layers, press SHIFT + 1KEY .. SHIFT + 9KEY .
- To toggle the visibility of layer 10 without affecting the visibility of the other layers, press SHIFT + 0KEY .

- To toggle the visibility of one of layers 11 .. 19 without affecting the visibility of the other layers, press ALT + SHIFT + 1KEY .. ALT + SHIFT + 9KEY .
- To toggle the visibility of layer 20 without affecting the visibility of the other layers, press ALT + SHIFT + 0KEY .
- To make all layers visible at once, press ~ . Press ~ again to return to your previous layer visibility setting.

Note to AZERTY users:

On the AZERTY keyboard layout, the standard number keys are the &e""(-è_à keys. Do not use Shift unless you want to toggle visibility as explained below.

Holding down Shift while selecting a layer (by keyboard or mouse) will, instead of making only that layer visible, toggle the visibility. In this way, you can select combinations or to hide particular layers.

The key to press to select all layers at once differs by keyboard layout. It is:

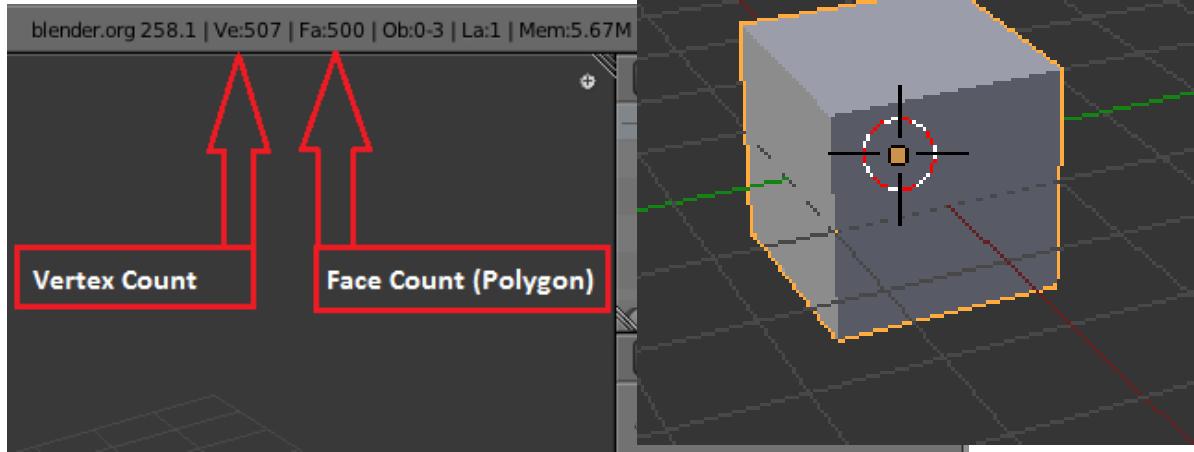
- ` (the key under Esc) on UK keyboards,
- ~ US,
- ö German, Swedish, Finnish and Hungarian,
- " Swiss German,
- æ Danish,
- ù AZERTY,
- ø Norwegian,
- Ñ Spanish,
- ç Portuguese,
- " Brazilian Portuguese,
- ò Italian, and
- ë Russian.

After pressing the aforementioned key, holding down Shift while pressing it again will restore the visibility settings you had *before* you made all layers visible.

When only one layer is selected, new objects are automatically assigned to that layer. When two or more layers are visible, new objects are assigned to the most recently visible layer.

1.17.9 Count Your Polys

If you want to count the polygons in your scene, the data is available in the Info Header.



As you can see in the above image, this scene has 507 vertices and 500 faces (polygons).

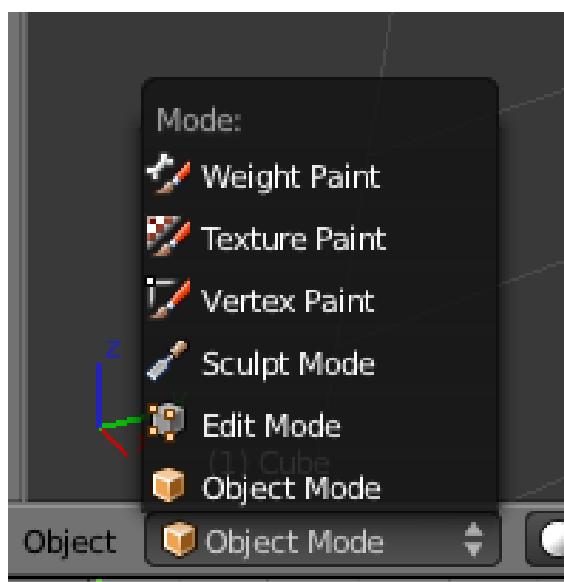
Cube selected in Object mode.

1.18 Object Mode

1.18.1 Introduction

In this module, you will learn some basics about operating in Object mode. This is normally the initial mode Blender is in when you open a new document. It is the mode where you operate on whole objects, rather than on their parts.

Many of the conventions involving selection and manipulation of objects or parts of objects apply to other modes as well, so this is a good place to become familiar with those conventions.



Open a new document, then confirm you are in Object mode by checking the mode menu.

Select the default cube by clicking on it with RMB . You will see it framed in an orange outline.

1.18.2 Object Origin

When you select an object, you will notice a round dot appears, normally in the middle of the object, the same orange-yellow as the rest of the selection.

This is the object's *origin*. It is the reference point for the object's local coordinate system. Certain kinds of edits to the object can cause this origin to end up at a position well outside the object. If that happens, operations like transformations applied with reference to the origin may not behave as expected. However, Blender has capabilities to deal with this. They will be explained when you need them.

1.18.3 Multiple Selections

You can select more than one object at a time. With the cube still selected, change your view until you can see both it and the default lamp. Select the lamp by clicking on it with SHIFT + RMB , so both it and the cube are selected. You will notice that the lamp takes on the orange-yellow colour, but the cube now has a more reddish highlight.

The active object is the last one selected. Other objects can be part of the selection, but the reddish-orange highlight indicates that they are not active. The Properties window shows properties for the active object, not the entire selection, although operations in the 3D view like moving and deleting objects will affect the entire selection. Some operations (like parenting, which you will learn about later) set up a special relationship between the

active object and the rest of the selection, so for these, the order of selection of objects becomes important.

You can remove the active object from the selection with SHIFT + RMB ; the small spot indicating the origin of the object's geometry stays highlighted in the yellow-orange colour, even though the rest of the object loses the selection highlight. If you do this to an inactive object, it will make that object active.

Pressing CTRL + I *inverts* the selection. i.e. it deselects what was previously selected, and selects everything else instead. It does not change the active object.

1.18.4 Selecting Obscured Objects

If multiple objects lie under the mouse, you can choose which one to select by clicking ALT + RMB : this will bring up a menu listing the names of the selectable objects.

Alternatively, you can add an object to the current selection, or remove it from the current selection, by clicking ALT + SHIFT + RMB and selecting it from the menu.

On Ubuntu 16.04 LTS, it appears that ALT + RMB has the same effect as RMB on a Window's title bar. But ALT + SHIFT + RMB does the trick of **Selecting Obscured Objects**.

1.18.5 Selecting Everything and Nothing

Pressing A does one of two things: if *anything* is selected, it clears the selection (i.e. selected objects are no longer selected). But if *nothing* is selected, then it selects *everything*. You will often see instructions to press A either once or twice, to ensure that either nothing is selected, or everything is selected.

1.18.6 Hiding Things

When working on a complex model or scene, things are likely to get cluttered, making it hard to see the specific part you're working on. It is possible to *hide* objects, so they no longer appear in the 3D view. Select the object(s) you wish to hide, and press H . This is purely a convenience for working in the 3D view, i.e. hidden objects remain unchanged when you render them.

Pressing SHIFT + H hides everything *except* the current selection. This is a quick way to remove the clutter and narrow the view to the objects of interest.

Pressing ALT + H brings back all hidden objects and selects them. If you lose track of what is hidden and what is visible, press this to bring everything back.

1.18.7 Local Versus Global View

Local view is another way of selectively hiding parts of the scene. Pressing NUM/ (\ for emulated numpad) hides everything that is not selected, and automatically zooms in or out as necessary so the selected objects fill the 3D view. Pressing NUM/ again, restores the items to the normal global view.

This differs from simple hiding with H in that a render done in local view only shows the objects currently visible in that view. In particular, if your lights are excluded from the local view, you are liable to see black blobs in place of your objects.

How do I determine the viewing mode? Look at the words in the upper-left corner of the 3D view. They indicate your current view orientation and perspective settings (e.g. "User Persp"). If the word "(Local)" appears at the end of the string, you are in local view. Otherwise, you are in global view.

1.18.8 Border Select (Box Selection)

A quick way to select many objects at once is with the Border Select (box selection). Press B to activate it. You will see a pair of dotted crosshairs appear centred at the current mouse position. Drag diagonally with LMB to mark a selection rectangle, then release the LMB . Everything within the rectangle will be added to the selection. If you didn't mean to engage box-selection mode, pressing ESC exits border select mode.

Alternatively, to remove things from the current selection, after pressing B , drag the selection rectangle with MMB . When you release the mouse button, everything in the drawn box will be deselected.

1.18.9 Circle Select (Brush Selection)

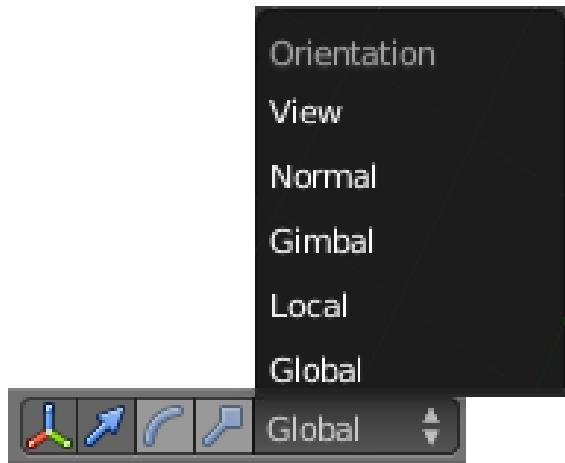
Another way to select several objects at once is with the Circle Select (brush selection), engaged by pressing C . In this mode, clicking or dragging on objects with LMB adds them to the selection, while MMB removes them from the selection. Thus the mouse becomes a brush that you can use to "paint" objects in or out of the selection.

The circle showing the size of the brush can be adjusted with the mouse wheel. This allows you to use a broad brush for selection of lots of objects at once, or a finer one for better control.

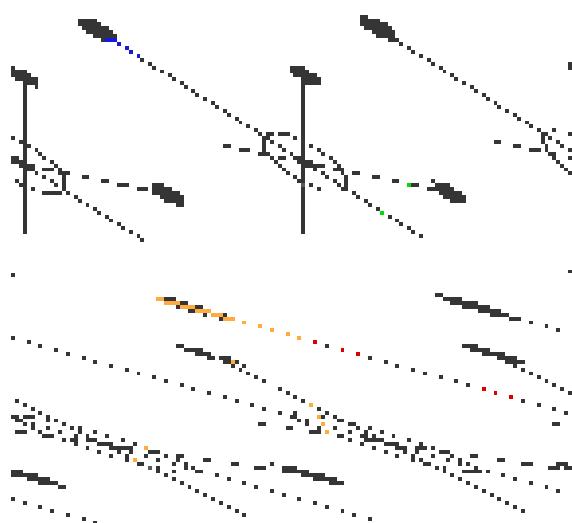
Clicking RMB or pressing ESC terminates Circle Select mode.

1.18.10 The Manipulator

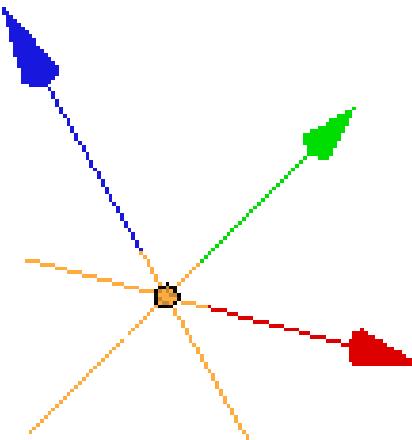
The manipulator appears in the middle of the selection. There are three kinds of manipulator as shown in the il-



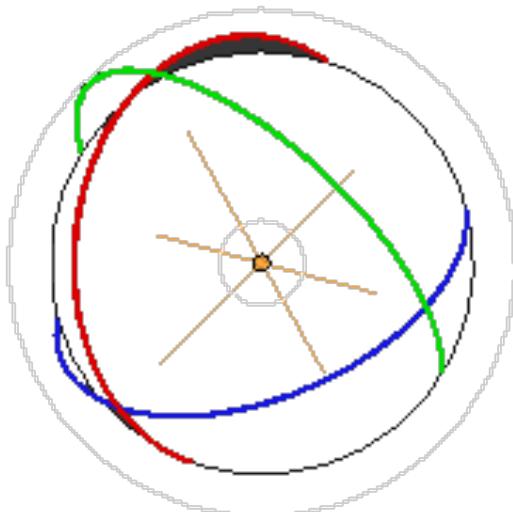
Manipulator transformation buttons & orientation menu



Manipulator—scaling



Manipulator—translation



Manipulator—rotation

lustrations. It can be used to apply translation (position changes), rotation and scaling (size changes) to objects. Its appearance changes according to which of these functions are enabled. You can click on the menu transformation buttons that appear when the manipulator is visible, to choose a single transformation, or shift-click to enable more than one simultaneously. You can toggle the visibility of the manipulator with **CTRL + SPACE**, or by clicking the menu button with the red, green and blue arrows.

Transform orientations: the “Orientation” menu governs how the axes of the manipulator are aligned, with the default “Global” corresponding to the global coordinate system. Other useful options are “Local”, which corresponds to the local coordinates system of *each object*, and “View”, which is always aligned to your view.

To demonstrate this, click on the camera with RMB so that it is the only object selected. Set the manipulator to do only translations (blue arrow button is selected in menu), and ensure the orientation is set to “Global”. Drag any of the manipulator’s coloured arrows with LMB to move the camera in the corresponding direction.

Now switch the orientation to “Local”. You will see the manipulator arrows re-orient themselves. Note that the Z-direction (blue arrow) is now in the direction of the camera view. The local co-ordinates of the camera have the optical axis of the camera running along the Z axis. By default, that is pointing towards the cube object.

The cube, by default, has its own local Z-direction running vertically.

With the manipulator orientation still set to Local, add the cube to the selection with **SHIFT + RMB**. You will see the manipulator move so it is in the centre of the selected objects. It is now between the camera and the cube. Now if you drag the manipulator Z axis arrow with LMB, each

object will move along its *own* version of that axis. The camera moves towards or away from the cube and the cube rises or falls.

Switch the orientation to “Global”, and try dragging a manipulator arrow again. This time, both objects will appear locked together and will move in the same direction, along the same (global) axis.

1.18.11 Transformation Hotkeys

The manipulator is not the only way to apply transformations to objects. That can also be done via keyboard shortcuts.

Hide the manipulator to reduce clutter. Select the cube, and only the cube, with RMB . Now press G to Grab the object. The selection outline around the object turns white, as it did when you were dragging with the manipulator, except this time, you didn’t press any mouse buttons. Now move the mouse without pressing any buttons, and you will see the object move along with it. Press LMB or ENTER to terminate the movement and leave the selected object at the new position, or RMB or ESC to cancel the operation and leave the object at its original location.

Similarly, use R to Rotate the object, and S to Scale it.

You can *constrain* the movement to particular axes by pressing the appropriate axis key. For example, press G to start moving the cube again, then press X and you will see a bright colored line appear parallel to the global X-axis. Now when you move the mouse, the cube will move along only that colored line. Similarly Y and Z constrain movement to the Y and Z axes respectively. The colored lines that appear are a brighter reddish, green or blue that correspond to the red, green or blue lines for the X, Y or Z axes, respectively.

Transform orientations: to constrain the transformation to a different set of axes, press the constraint key twice. The coordinate system used depends on the selection in the Transform Orientation menu:

- Local or Global — the transformation happens in the object’s local coordinate system.
- View — the transformation is aligned to view coordinates.

For example, with the default “Global” selection from this menu, select the camera with RMB , press G to move it, then press Z twice, and you will see the coloured line orient itself along the direction of view of the camera.

The axis constraints also work with scaling, and rotation (which only happens *around* the specified axis).

You can also constrain movement and scaling to happen along two axes, but not the third one, by holding down SHIFT when typing the axis constraint. For example, G

followed by SHIFT + Z will constrain movement to the global X-Y plane (i.e. any direction *except* along the Z-axis). To constrain movement to the *local* X-Y plane, type the constraint twice: G SHIFT + Z SHIFT + Z .

Here’s a summary of what the transformation hotkeys do, with and without constraints:

In addition, the hotkey sequence R R enables *free rotate*, i.e. the object can rotate around all three axes as you move the mouse.

Transforming by Numbers

Sometimes you need to position things accurately, using calculated numbers, instead of estimating by eye. Blender can do that too. Simply type the number after the transformation hotkeys before pressing ENTER to confirm the operation. For example, G X 1KEY ENTER will move the selection by 1 unit in the positive X direction. G X -KEY 1KEY ENTER will move by 1 unit along negative X. Decimal points are allowed, thus S 0KEY .KEY 5KEY ENTER will scale the selection by a factor of 0.5, or 50%.

Rotation works similarly, using degrees clockwise around the selected X, Y or Z axis.

Transformation Menu

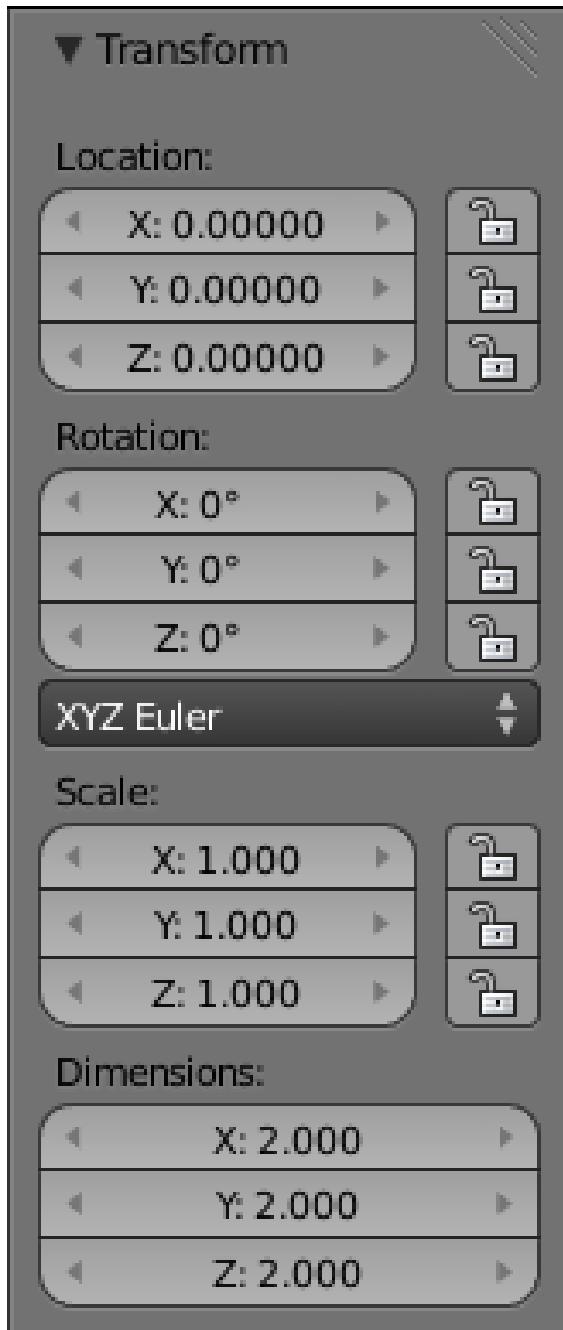
Yet another way is shown at right, in the Transform panel that appears at the top of the Properties shelf (press N to toggle its visibility at the right side of the 3D view). Here you can see the existing transformations values. You can drag the sliders to change them, or click on them and enter new values.

1.18.12 Choosing the Pivot Point

When you do a scaling or rotation operation, you can choose the pivot point, which is the central origin point that remains unaffected by the operation. By default this is the “Median Point”, or centre point of the selection, but the Pivot Point menu lets you choose some other options. For example, select both the cube and the camera, and rotate them (R). By default they will rotate around their common centre. Now go to the Pivot Point menu and choose “Individual Origins” and rotate your two selected objects with R again, and you will see each one now rotates about its *own* centre, rather than the common one.

Another useful pivot option is “3D Cursor”, which places the transformation origin at the 3D cursor location.

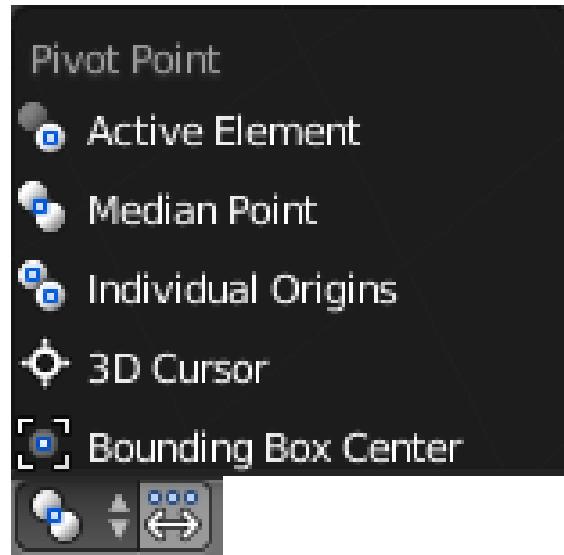
Finally, the button with three dots and a double-headed arrow immediately to the right of the one that pops up the Pivot Point menu is titled “Manipulate center points.” Selecting this means transformations do not rotate the actual objects themselves, only their positions. To see the



effect, you need to choose a pivot point that is not the object's origin. Now try rotating the object. You will see its centre describes an arc around the pivot point, without changing the object's orientation. Think how the seats in a Ferris Wheel rotate around the wheel's pivot, yet still maintain their orientation.

Similarly scaling will change the distance between the chosen pivot point and the object's origin, but will not affect the size of the object itself. A bursting firework scales rapidly in this way.

Why can't I rotate or scale objects? One pitfall you might encounter is that you select an object, try rotating with R or scaling with S, and nothing happens, though



moving with G still works. It's quite likely you have the "Manipulate center points" button active when you didn't mean to. Check if it's active, and click it to deactivate if so.

Hotkeys — there are keyboard shortcuts for all the above options:

1.18.13 Basic Camera Technique

The camera view NUM0 is very useful for making adjustments to your camera while getting continuous feedback on how the render will look. This view shows a framing rectangle covering the area that will appear in the render, surrounded by a *passepourtout* which gives a darkened view of the surrounding part of the scene. You can use the mouse wheel to zoom in and out, adjusting how much of your view is the rendered area and how much is passepourtout.

In this view, use RMB on the framing rectangle to select the camera, and it will show the usual orange-yellow highlight. The manipulator will not appear even if enabled, so you must use the transformation hotkeys to perform camera transformations.

Use G to move the camera around parallel to the view plane. Since the view stays locked to the camera, you will see the scene move in the direction *opposite* of what you might expect.

The camera's local Z-axis lies along its direction of view. This allows useful operations like G Z Z to move the camera in or out without affecting the direction in which it's pointing. Also the X axis runs left to right in the camera view so rotating around X R X X will adjust the up-and-down *pitch* angle. Rotating around the vertical Y axis R Y Y will change the *yaw* (left-right) angle, and you can rotate around the optical axis of the camera using R Z Z to produce an effect of *rolling* the view around the visual

axis.

Another useful technique is to position the 3D cursor at a point of interest, set the pivot point to the 3D cursor, then rotate the camera about a global axis, like the global Z-axis (R Z), to adjust the angle of view while keeping the same objects in view, and without altering the distance of the camera from the point of interest. In real life you'd get that effect by walking in a circle around your subject with your camera mounted on a *Steadicam* rig.

Scaling the camera object changes its size as shown in the 3D view, but has no effect on the actual render. Regardless of what axis constraints you try to apply, the camera object will always scale uniformly along all axes.

You can also use Fly mode SHIFT + F in camera-view mode to fly around the scene, taking the camera with you.

Another choice for moving your camera in Camera View is to bring up the Properties panel (N) and, in the View section, tick the box next to Lock Camera to View. Now you will be able to use the MMB to “move objects” just as you move things around in other views such as the 3D view. Holding down the MMB and dragging will rotate, Shift + MMB will allow you to “move the object” around in the view (panning), and the scroll wheel will allow you to “move the object” closer or farther from the camera. You are actually moving the camera with these manipulations and not the object(s) themselves.

1.18.14 Adding/Removing Undo/Redo, Repeat

Objects,

Select the cube with RMB again. Press either X or DEL and, after confirming the popup, the cube disappears! It has been deleted from your scene. Unlike mere hiding, it really has disappeared. Press CTRL + Z to undo your last operation, and it reappears.

Click with LMB to position the 3D cursor away from the default cube. Press SHIFT + A to bring up the Add menu, go to its Mesh submenu, and add another cube to the scene. Again, undo with CTRL + Z , and you are back to a single cube again.

Now press CTRL + SHIFT + Z : this will undo the undo, and *redo* the last operation you undid, bringing back the second cube.

Try adding a third cube. Now CTRL + Z should undo that and take you back to two cubes, and pressing CTRL + Z again should undo the addition of the second cube, taking you back to one. Try CTRL + SHIFT + Z at this point to restore the second cube, then CTRL + SHIFT + Z again to restore the third one.

Blender remembers up to the last 32 things you did (depending on the limit set in your user preferences) in its *undo stack*. You can go backward and forward through it with CTRL + Z and CTRL + SHIFT + Z .

Sometimes you want to perform an action repeatedly. To repeat the last action, type SHIFT + R .

1.18.15 Assigning Layers

Earlier, you learned about showing and hiding layers in the 3D view. To assign layers for selected objects, press M . The same keyboard shortcuts apply here as when choosing which layers to display, i.e. 1KEY for only the first layer, 2KEY for only the second etc, SHIFT + 1KEY to include/exclude the first layer and so on.

After assigning an object to a different layer, it disappears! If this happens to you, it's because the layer(s) you assigned to the object, and the layer(s) you currently have visible in the 3D view, have nothing in common. Simply change the visible layers to include at least one of those you assigned the object to, and it will reappear. For example, if currently only layer 1 is visible, and you assign an object to only layer 2, it will disappear, but reappear when you change the visible layer to layer 2.

1.18.16 Object, Action, Settings

Bring up the Add menu again (SHIFT + A). This time, add a new cylinder mesh to the scene. Look to the left of the 3D view, in the Tool Shelf (toggle its visibility with T if it's not visible), at the bottom you should see a new panel has appeared, titled “Add Cylinder”. Near the top of it is the “Vertices” number, initially defaulting to 32, which gives a fairly round-looking cylinder. Reduce it to 6, and adjust the view as necessary to get a good view of your “cylinder”, and you will see it is now a hexagonal prism. Change the number of vertices to 3, and it becomes a triangular prism.

This is an example of an important user-interface convention that runs right through Blender: first you select the object you want to perform an operation on as appropriate (not applicable here because we are creating a new object), then you perform the specified action with some default settings, and finally you adjust the settings to give the exact result you want. This way, instead of getting a popup *before* the action is performed, into which you have to put the right settings and hope they will give the right result, you get to interactively adjust the settings and immediately see the results, without having to continually redo the operation and deal with popups.

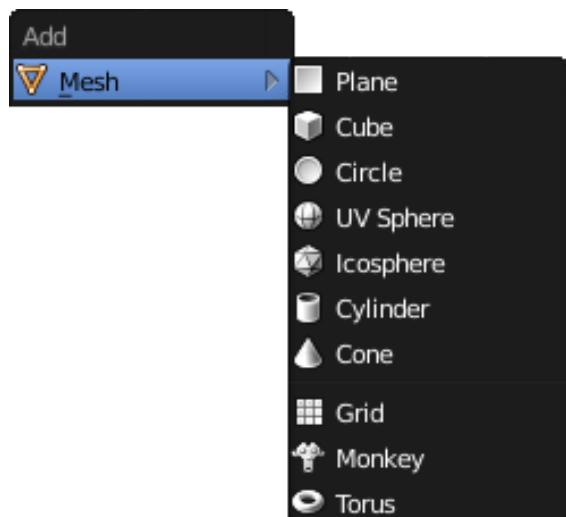
Chapter 2

Unit 2 - Basic Modeling and Shading

2.1 Meshes and Edit Mode

A mesh is one of the most important and frequently-used object types in Blender. While there are other types of objects that can be used to model parts of a model or scene (text, NURBS patches, etc.), they often get converted to meshes at some point anyway, because it is the object type that offers the greatest amount of detailed control. And as it happens, Blender offers more functions, both built-in and available as addons, for dealing with meshes than for any other object type.

Edit mode is the mode in which you make changes to the internals of the *active* object. Not every object has an Edit mode (e.g. cameras), and the details of what you can do in Edit mode vary between the object types where it is available. This module specifically covers Edit mode for mesh objects.



scratch.

2.1.1 What Is a Mesh?

A *mesh* is made up of one or more *vertices*; each *vertex* is just a point in space. A pair of vertices can be joined by a straight line called an *edge*, and a complete loop of edges can be filled in to form a *face*.

It is the faces that make up the visible surface of the object. The edges and vertices are essentially geometrical “scaffolding” necessary to hold the object together.

A face must have three or more sides (edges). Prior to version 2.63, only three or four sides were allowed, so faces had to be *triangles* or *quadrilaterals* (usually abbreviated to *quads*) respectively. Starting with v2.63, and the introduction of the BMesh architecture, that restriction has been lifted. You can have faces with 5 or more sides, but you will usually find that things work best if all faces, as far as possible, are quads. Particularly when constructing a model for animation purposes.

Blender’s Object-mode “Add” menu (SHIFT + A) contains a Mesh submenu with a collection of pre-made mesh objects. Think of these as starting points. They make building your own objects easier by enabling you to modify an object that is an approximation of the form you want instead of having to construct a mesh entirely from

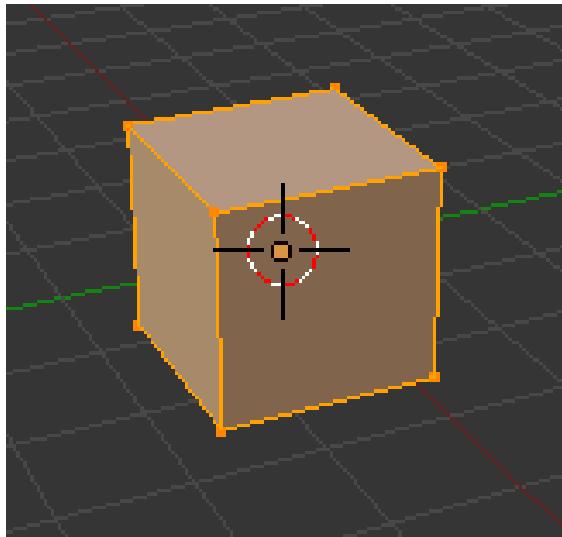
2.1.2 Introduction to Edit Mode

Open a new Blender document. Hide the manipulator if it is visible (CTRL + SPACE). You should be in Object mode. Click with RMB on the default cube to ensure it is selected and the active object.

You can switch modes using the mode menu, as you previously learned. However, because switching between Object mode and Edit mode is such a frequent operation, it has a keyboard shortcut: TAB . Do this now, and you should see the appearance of the cube change, as shown at right. The mode menu should also update. Press TAB again, and you should be back in Object mode. Press TAB once more before continuing, to ensure you are in Edit mode.

Note the following features of the cube, and how they relate to the description of a mesh above:

- The dots at the corners are the vertices.
- The lines joining them are the edges.
- The filled areas bordered by the lines are the faces.



The default cube in Edit mode

2.1.3 Selection Modes



Select mode buttons in a 3D View header, showing Vertex select mode active. (Found below the edit area.)

In Edit mode, the header (it's at the bottom) of the 3D View window changes to show the selection-mode controls. If you hover over each of the buttons in the group of three, you will see they represent vertex-select, edge-select and face-select respectively. You can shift-click to enable more than one at a time.

In vertex-select mode, you select a single vertex by clicking on it with RMB, and select more than one by shift-clicking on additional vertices with RMB. Shift-clicking with RMB on an already-selected vertex will deselect it.

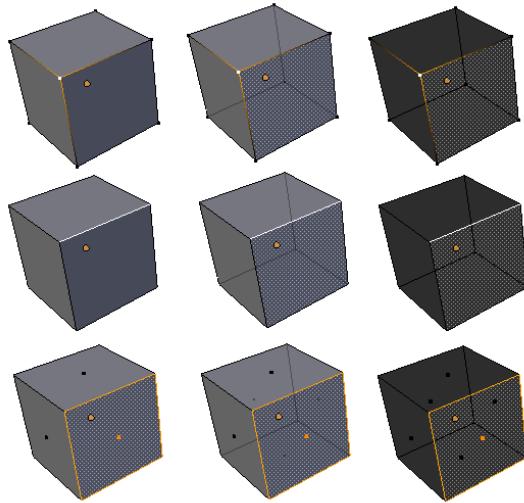
Pressing A will select *all* vertices if none are currently selected, otherwise it will *unselect* all vertices.

Edge-select mode works in a similar way, except with edges instead of vertices. Similarly, face-select mode will allow you to select and unselect faces.

The single button immediately to the right of these three is titled “limit selection to visible”. When it is active (the default), the mesh object being edited is displayed as *opaque* which means that vertices, edges or faces on the side away from you are hidden and cannot be selected. Click this button, and the object becomes translucent, allowing clicking *through* front faces to select parts of the

mesh behind them.

Another useful display mode for working in the 3D view is wireframe, which can be selected from the Viewport Shading menu or toggled with the Z key. In this mode, the faces become transparent, almost invisible, and the edges and vertices are displayed more prominently.



Here is what these various selection and display modes look like in combination: in the first row, a single vertex is selected. In the second row, a single edge, and in the third row, a single face. In the first column, limit-selection-to-visible is enabled. In the second column, it is disabled, and in the third column, wireframe mode is enabled.

Selection Mode Hotkeys

You can also switch selection modes with CTRL + TAB . In the menu that appears, you can switch to a single selection mode by selecting it with the mouse or up/down-arrow keys and pressing ENTER or LMB . But if you press SHIFT + ENTER or SHIFT + LMB that toggles the enabling of only that selection mode, without affecting the state of the others, as does shift-clicking on the icons above.

In common with other Blender popup menus, you can quickly select an item from the CTRL + TAB menu and immediately confirm by pressing one of 1KEY , 2KEY or 3KEY to select the first (vertex), second (edge) or third (face) item in the menu. Or, SHIFT + 1KEY , SHIFT + 2KEY and SHIFT + 3KEY while the menu is up, will toggle the enabling of vertex, edge, and face-select modes respectively.

2.1.4 Multiple Selections

You can use SHIFT + RMB to select multiple items, and CTRL + I to invert the selection, just like in Object mode.

Only here, the “items” are vertices, edges or faces, depending on the selection mode in effect. The active (last-selected) part is shown in white, while the rest of the selection (if any) is drawn in the usual orange-yellow colour.

As mentioned above, A works similar to the way it works in Object mode, only instead of applying to everything, it applies to all parts of the object being edited.

2.1.5 Hiding Things

H , SHIFT + H and ALT + H work in a way analogous to their behaviour in Object mode. Again, instead of applying to everything, they apply to all parts of the object being edited.

Remembering What’s Hidden: If you switch out of Edit mode with some parts hidden, they will reappear, then disappear again when you re-enter Edit mode, i.e. each object remembers what was hidden when you last edited it.

2.1.6 Local Versus Global View

You can toggle local/global view in Edit mode, as you can in Object mode. However, instead of narrowing the view to one or more selected objects, it narrows it to just the object being edited.

2.1.7 Border Select (Box Selection) & Circle Select (Brush Selection)

B and C work analogously to the way they do in Object mode, i.e. you select multiple items by drawing a box or by “painting” over them.

2.1.8 Select More, Select Less

Edit mode has some additional selection capabilities. To demonstrate them, let’s use something other than the default cube, for a change. TAB back to Object mode, and delete the cube. Now add (SHIFT + A) a Grid object. TAB into Edit mode, and you will see that the grid is made up of 9×9 faces, or 10×10 vertices. Initially they will all be selected. Use A to unselect them then C to brush-select a few vertices in the middle. End brush-select mode with RMB or ESC . Now watch what happens to the selection when you press CTRL + NUM+ (select more). Additional vertices adjacent to those already selected are added to the selection. Now try CTRL + NUM- (select less), and you will see the vertices on the edge of the selection are removed from it.

2.1.9 Manipulator, Transformation Hotkeys, Pivot Point

All of these are available for use in Edit mode as they are in Object mode, except for the “Manipulate center points” button.

Note that scaling vertices scales the distances between them. The vertices themselves have no size, so they do not get larger or smaller. Similarly, rotating vertices only changes their direction relative to the pivot point, since a featureless point itself has no orientation.

Transform Orientations

The Global, Local and View options in the Transform Orientation menu apply in Mesh Edit mode as they do in Object mode. In addition there is Normal mode, where the transformation axes are aligned relative to the selection:

- If a single face is selected, the X and Y axes are aligned along the face, while the Z axis is aligned perpendicular (normal) to the face.
- If a single edge is selected, the Z axis is aligned along the edge, with the X and Y axes perpendicular to it.

Other selections are also possible. Feel free to investigate their behaviour for yourself.

Thus, with a single face selected, G Z Z will move the face along its normal.

In addition, it is possible to define the current Normal transformation orientation as a custom orientation for use in transforming other vertices and even other objects. To do this, you need to go to the Properties Shelf, which is made visible on the right of the 3D view with N . Near the bottom is the Transform Orientations panel which contains a Transform Orientation menu which looks the same as the one in the header of the 3D View window, except it also has a “+” button next to it. Click the “+” and the current Normal transformation orientation will be added to the menu initially labeled “Vertex”, “Edge” or “Face”, depending on what is currently selected (and with a unique numeric suffix added if there is already a custom orientation defined with that name). Now if you look in either Transform Orientation menu, you will see a new selectable item, in a separate section above the five standard items.

With your new orientation option selected, an editable text field will appear in the Transform Orientations panel, allowing you to change the name if you wish, and there is also a “X” button allowing you to delete the orientation item when you no longer need it.

With your new orientation option selected, you can now select a different part of the object, or even TAB into Object mode and select some other object. The manipulator

and the doubled axis hotkeys will now align their transformations along this custom orientation. This is handy, for example, for aligning objects to a sloping plane.

2.1.10 Proportional Editing

When trying to produce natural, organic shapes, moving vertices one by one gets tedious. To produce smoother looking shapes, you need a mode where unselected vertices close to the selection also get some movement. In contrast to the sharp distinction between selected vertices which are moved and unselected ones that remain in place, there is a gradual transition from one to the other.

This is where proportional editing comes in. If you select “Mesh” in the header and examine the pop-up menu, you will see two submenus, titled “Proportional Editing” and “Proportional Editing Falloff”. The former toggles the mode on and off, the latter controls the falloff function choice. There is also an icon for “Proportional Editing” in the header of the default 3D view. Look to the right of the “Limit Selection to Visible” (on edit mode). The hot key is the letter O. Pressing O will toggle between Enable and Disable. Pressing SHIFT + O will change the Falloff type.

The “Proportional Editing” submenu has 4 options: “Disable”, “Enable”, “Projected (2D)”, and “Connected”. Do you still have the Grid object you created in the “Select More, Select Less” section above? If not, add a fresh Grid object. Switch to Edit mode, and ensure that just a few vertices in the middle are selected. Enable proportional editing, and now use G to move the selected vertices. You should notice 2 things:

- unselected vertices near the selected ones also move, and
- there is a white circle enclosing all the vertices that undergo any movement.

Try using the mouse wheel while moving the vertices, and you will see the white circle grow or shrink, and the proportional region of influence will grow or shrink correspondingly.

Try different falloff functions in the “Proportional Editing Falloff” submenu. Some ensure the mesh stays smooth and curvy, others give a more angular effect, etc.

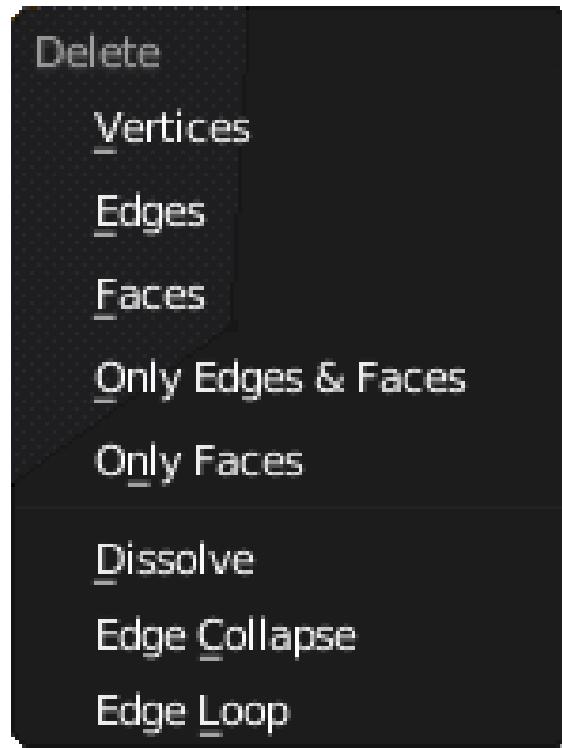
The third option to “Disable” and “Enable” proportional editing is “Projected (2D)”. This view is similar to enabled, except depth is ignored. The radius of the influence region is applied to the mesh two dimensionally.

The fourth option to “Disable” and “Enable” proportional editing is “Connected”. This one makes a difference in more complicated meshes, which might have folds or concavities in them. In this situation, “Enable” affects all vertices within a particular distance of the selected ones,

while “Connected” only measures the distance *via connected edges*, rather than directly through space. This lets you move one part of the mesh without affecting another part which might be located nearby purely as a result of a fold.

Of course, proportional editing works with scale S and rotate R operations as well.

2.1.11 Deleting Things

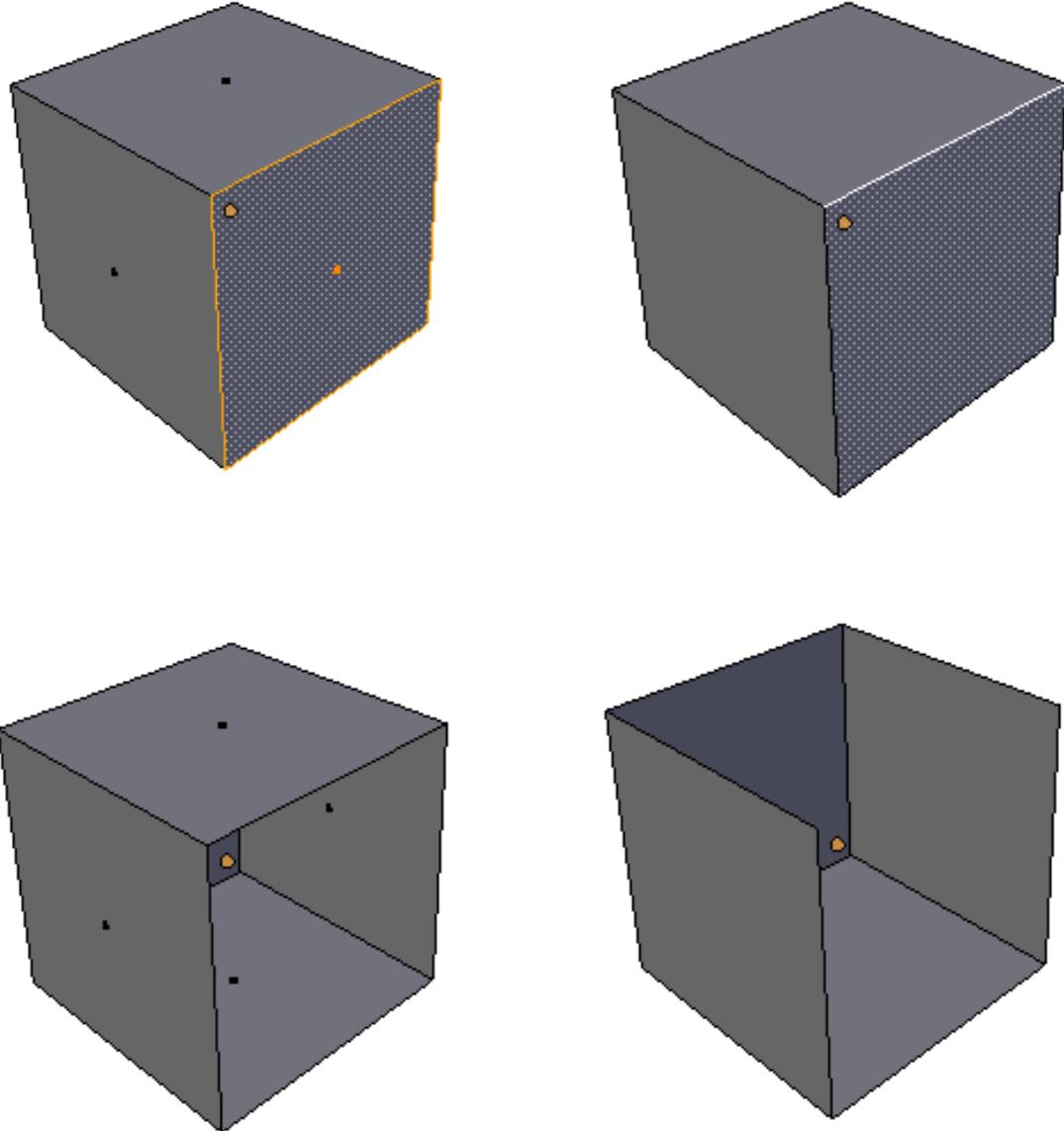


Now let's try deleting parts of a mesh. This is the menu that comes up when you press X or DEL when editing a mesh. For now, we will concentrate on the first three items.

First, go into face-select mode. Select one face of the default cube, press the delete key, and select “Faces”. As shown in the screenshots, the selected face should disappear.

Use CTRL + Z to undo your previous deletion. Now go into edge-select mode. Select one edge this time. Press delete again, this time select “Edges”. As the screenshots show, the selected edge disappears, but *the faces bordering that edge also disappear*. Faces cannot exist without their bordering edges!

Use CTRL + Z to undo your previous deletion again. Go into vertex-select mode. Select one vertex, make sure it's the one closest to you so you get the best view of the effect. Press delete, and select “Vertices”. Not only does the selected vertex disappear, but also the edges connected to that vertex. Edges cannot exist without their



endpoint vertices. And since those edges disappeared, the faces dependent on them for their borders were deleted as well.

So, to recap:

2.1.12 Undo/Redo

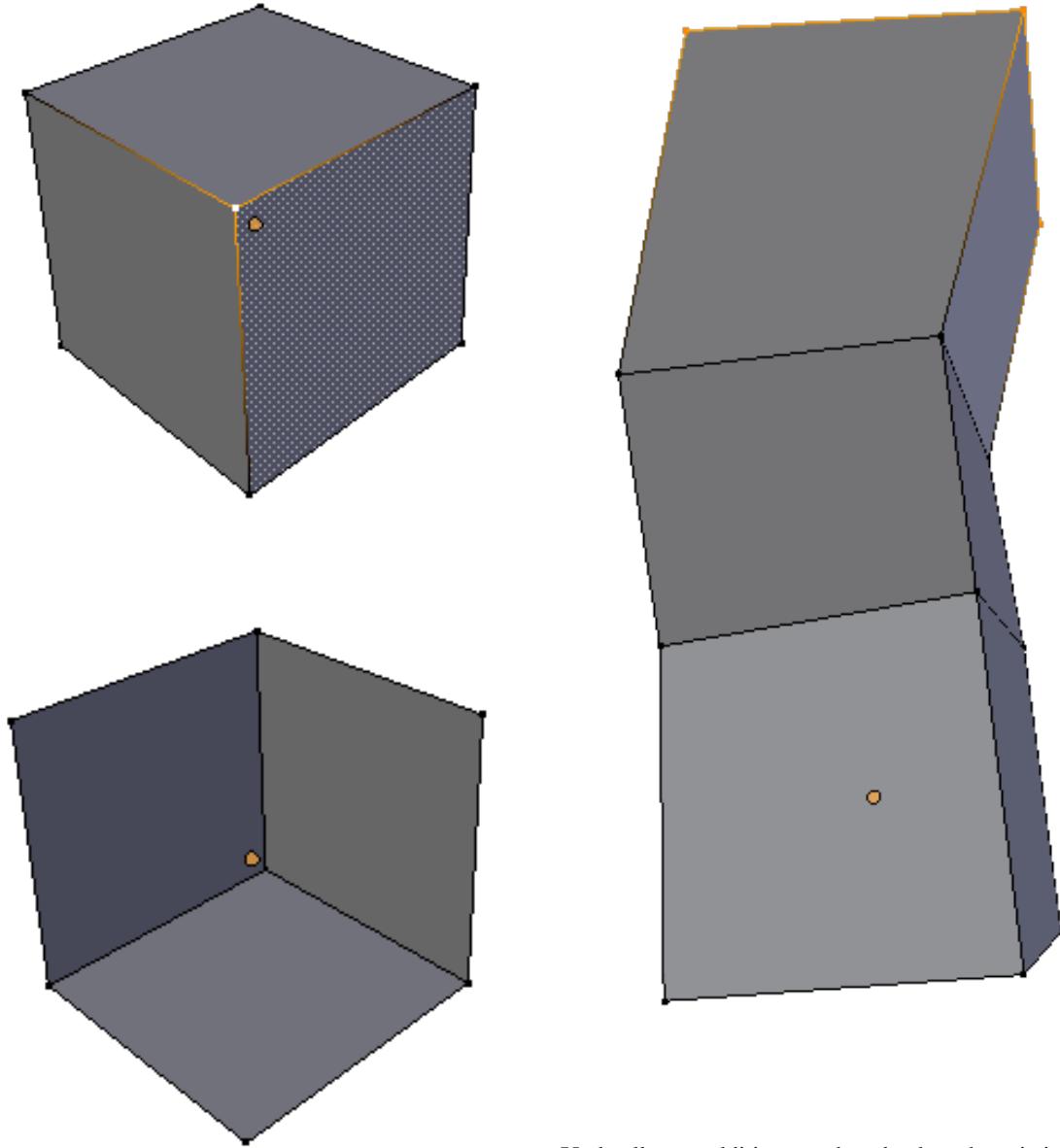
You can undo your last Edit-mode operation with **CTRL + Z**, and undo your undo with **CTRL + SHIFT + Z**, similarly to Object mode. However, Edit mode maintains its own undo stack, separate from the Object-mode stack. To undo/redo an Edit-mode operation, you must be in Edit mode, not Object mode.

2.1.13 Adding Things

Back to the default cube and Edit mode. Ensure you are in vertex-select mode with nothing selected. Do a **CTRL + LMB** somewhere near the cube. Do you see a little orange-yellow dot appear where you clicked? You just added a new, unconnected vertex to the mesh.

Undo your addition (**CTRL + Z**). Select an existing vertex with **RMB**. Now **CTRL + LMB** to add a new vertex again. You will notice that it is connected to the previously selected vertex by a newly added edge as well. Since the newly added vertex is now the selected vertex, doing **CTRL + LMB** again at another position, and so on repeatedly, lets you construct a whole chain of new edges. But what good are edges and vertices without faces?

To construct a face, you need a closed loop of edges. To close a loop of edges, select all the vertices in the chain,



and press F . That will add another edge joining the first and last vertex into a complete loop of edges and fill in the loop with a new face.

If you want to close the loop without filling in the face, select only the first and last vertex in the chain before pressing F . Since only two vertices are selected, a new face will not be added, i.e. it will only add an edge joining the two vertices.

You can also *extrude* new sections of mesh with a single click in this way. Try selecting two adjacent corner vertices of the cube (i.e. two joined by an edge). Now CTRL + LMB near them, and you will see you've created *two* more vertices, joined to the previous two by a new face. CTRL + LMB again, and you can construct a whole sheet of new mesh in this way.

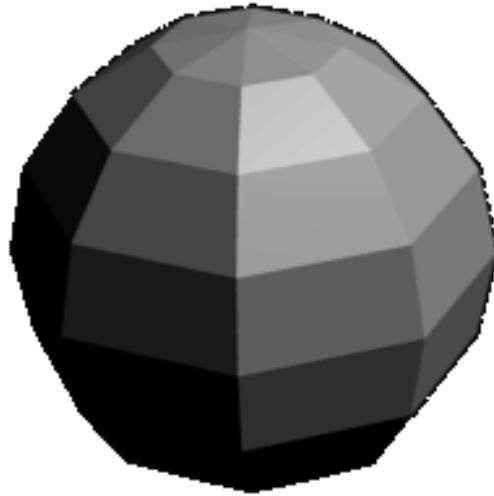
Undo all your additions, and get back to the pristine cube. Now select all four vertices of a single face. CTRL + LMB near your selection, and you should have four new vertices (corresponding to the original four you selected), plus a new face connecting them, plus *four* new faces connecting them to the original four. (You may not have noticed it, but the face formed from the original four vertices has been removed as well.) Another CTRL + LMB does the same thing again. So with just a few clicks, you started with a cube and ended up with something (see at right) that is starting to resemble — who knows? A square-cross-sectioned piece of some wonky-looking pipe, perhaps?

2.1.14 Simplifying Things

It is possible to remove vertices without leaving holes behind in the mesh, by merging two or more vertices into one. Select the vertices you want to merge, and press ALT

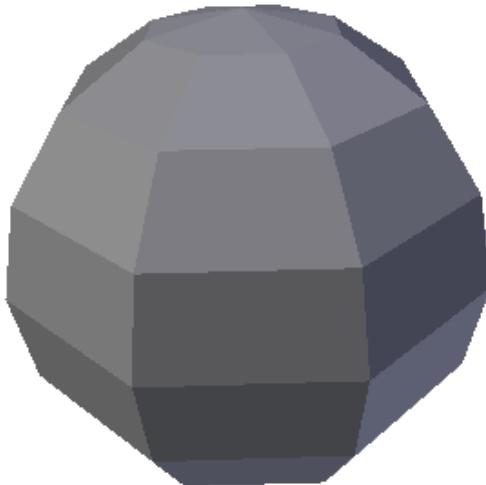
+ M ; a menu will pop up with some options, including whether to position the resulting vertex in the middle of the ones being merged, or at the position of the first or last one you selected. The resulting vertex inherits all the edges that were connected to the vertices being merged, as well as the faces connected between those edges.

Sometimes an operation creates duplicate vertices in exactly the same positions, or very close together. You can merge these *en masse* by ensuring you have selected all possible candidate vertices (e.g. the whole mesh), bringing up the Vertex Specials menu (W) and selecting the “Remove Doubles” item. Look for a message saying “Removed n vertices” to show briefly in the Info window. If n is 0, nothing was done. If you look in the lower left part of the Tool shelf, you will see a “Remove Doubles” panel has appeared, with a “Merge Distance” slider that governs the maximum distance allowed between vertices that are merged. Change this value as appropriate (either by clicking on the left and right arrows, or by clicking and typing in a new value and pressing ENTER), and the Remove Doubles operation is immediately redone. A new message will indicate how many vertices were removed. Simply keep adjusting the value until you are satisfied you haven’t removed too many or too few vertices.



replaced with the UV/Image Editor view, showing the rendered image, as at right. Press F11 to return to the 3D view .

2.2 Normals and Shading



Open a new Blender document. Delete the default cube, and add a “UV Sphere” mesh. In the “Add UV Sphere” (Picture shows Ico Sphere) panel which appears at the lower left of the Toolshelf (press T to make the Toolshelf visible if it’s not), set both the Segments and Rings to a low number, e.g. 8. The result will be very angular, as shown to the right, not round like you would expect a sphere to be.

Press F12 to do a quick render. The 3D view will be

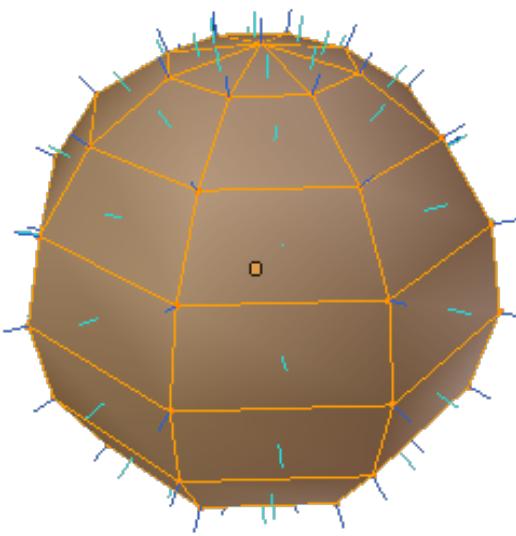
Select the sphere object (RMB). Now look in the Toolshelf for the shading buttons:



, and click on “Smooth”. Now try a new render with F12 .

As you can see, the *surfaces* of the object look a lot smoother and curved, even though the *outline* or *silhouette* is just as angular as before.

Return to the 3D view with F11 . Ensure the UV sphere object is selected, and you are in Edit mode. Bring up the Properties shelf at the right side of the 3D view with N if it’s not already visible. Look for the



Mesh Display panel, and find the settings for Normals.



If you check both icon boxes, the display of the UV sphere should change to look something like the image to the right.

Those spiky little lines are the normals; the green ones in the middle of each face are face normals, the blue ones protruding from each vertex are vertex normals.

In the physical theory of light, the *normal* is a line perpendicular to the surface of the object the light is hitting. When your eye (or the camera) C is positioned on a plane through the normal of a particular surface observing a certain surface point P illuminated by a coplanar light source L , a specific amount of light will be reflected and hence be registered by the camera depending on the physical characteristics of the surface. The observed intensity of reflected light is at a maximum if the angle $C-P-L$ is divided into two equal halves by the normal.

In the real world, a lot of surfaces are curved or otherwise not flat. But a mesh can only be made up of straight edges and flat faces. So how can it represent an object with a curved surface?

When you added the UV sphere to your scene, you had the option of specifying how many segments and rings it was made from. The more of those present, the closer the geometry approximates a curve. However, the more there are, the longer the render will take, and the more memory the model will consume to hold information about all the extra vertices, edges and faces.

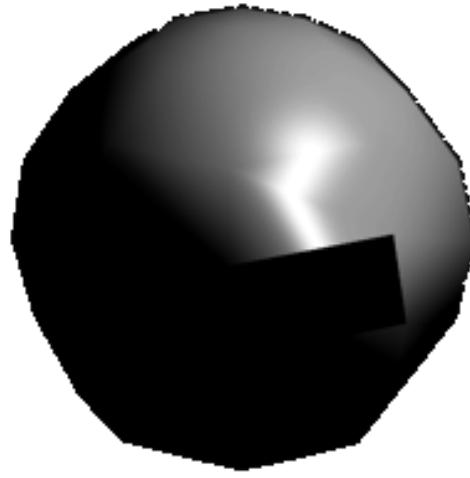
Which is where that “Smooth” shading button you clicked comes in. It applies a trick called *Phong shading*. Instead of doing the lighting calculation based on a normal for each face as the physical theory says you should, it starts with a normal assigned to each vertex, and interpolates

the normal at each point on a face from the vertex normals at its corners, based on the distance at that point to those corners. The result fools the eye into seeing curved, rounded surfaces where there aren’t any.

This completely violates the laws of physics. To start with, how can you define a “normal” which is perpendicular to a *point*? But as you can see, the results look rather good, with relatively little extra computation involved, much less than actually generating all the extra geometry.

As you learn more about computer graphics, you will come across more tricks like this. Physically accurate modelling is still very difficult to do, even with modern computers, and the results may not look all that good. But by adopting a bit of lateral thinking that goes completely against physics, we can often, ironically, come up with much more realistic-looking results.

2.2.1 Not So Smooth?



If you have been adding lots of vertices, edges and faces to your mesh, you may end up with discontinuities in smooth shading causing unsightly blotches, as shown to the right.

Assuming your mesh is constructed properly (e.g. no edges and faces cross each other in physically impossible ways), the most likely reason for this is the normals in the newly added vertices and faces are pointing the wrong way. To fix it, select the troublesome part of the mesh (or select the whole thing) in Edit mode, and press **CTRL + N** to recalculate all the normals. Re-render the scene to confirm the shading discontinuity has disappeared.

2.3 More Mesh Editing Techniques

You previously scratched the surface of the tools that Blender provides for editing meshes. This page will in-

troduce more of them.

2.3.1 Adding More Mesh Pieces

Start with the default cube again. Select it and TAB into Edit mode. Press SHIFT + A to bring up the Add menu. Instead of all the submenus with all the objects you could add in Object mode, you will see only a single menu containing only mesh objects. Select another cube, and use G to move it away from the first cube.

If you TAB into Object mode, you will see that the two cubes look like separate, disconnected objects, but they are in fact one object, and cannot be selected separately in Object mode. You can TAB back into Edit mode, and make connections between the vertices of the two cubes, which you cannot do with separate objects. Therefore:

2.3.2 Linked Selections

If you have some part of a mesh selected, pressing CTRL + L will select all other parts of the mesh that are connected to the already-selected parts. In the above case of the object made up of two disconnected cubes, you can RMB on a single vertex of one cube, then use CTRL + L to select all the rest of that cube but not the other.

Another way to do linked selections is to simply move the mouse over some part of the piece you want to select, and press L to immediately select everything connected to that. Conversely, SHIFT + L will *unselect* everything connected to the vertex under the mouse.

2.3.3 Separating and Joining Meshes

You can separate a part of a mesh into its own object. The part you are separating doesn't have to be disconnected from the rest of the mesh. Simply select the part you want to separate in Edit mode, and press P , and in the menu that appears, choose "Selection". You will see the selected part immediately change to a reddish-orange highlight, indicating it is part of the object selection but not the active object.

Conversely, you can join two or more mesh objects into one. Select all the desired objects in Object mode, and press CTRL + J , and you will see them all immediately take on the orange-yellow highlight indicating they are *all* the active object. TAB into Edit mode, and you can confirm all are editable as part of the same mesh object.

2.3.4 Proper Extrusion

You previously discovered how to add whole new sections to a mesh with CTRL + LMB . Blender also has a proper Extrude function, which lets you do this with a bit more control.

Start with the default cube, as usual. Go into Edit mode. Select just the top four vertices. Press E to start extruding, and move the mouse roughly along the direction of the Z-axis. You will find yourself dragging out a whole new face formed from four new vertices connected to the ones you previously selected. You will notice also that the movement of the newly-added part of the mesh is automatically constrained to be parallel to the Z-axis. Press LMB or ENTER to finish the extrusion operation.

Deselect everything. Now try selecting another four vertices of the original cube, say making up a face pointing along the X-axis. Now if you extrude these, you will see that the extrusion is automatically constrained to move only parallel to the X-axis.

A quirk of the extrusion function is that if you press E and then immediately abort with RMB or ESC , *the additional mesh piece is still created*, but it is left in the same position as the original mesh. To *really* abort the extrusion, you have to undo it with CTRL + Z .

More Extrusion Options

ALT + E brings up the Extrude menu, which gives you access to more options, depending on what you have selected:

- "Region"—extrude the entire selected area as one, exactly equivalent to E .
- "Individual Faces"—if you have more than one face selected, then they are extruded separately. In particular, any edge common to two selected faces will give rise to two separate extruded edges, rather than one.
- "Edges Only"—extrudes only the edges; new faces are created only connecting the new edges to the existing ones, not between the new edges.
- "Vertices Only"—extrudes only the vertices; edges are created only connecting the new vertices to the existing ones, not between the new vertices, and no new faces are created.

2.3.5 Edge Loop Selection

Edge loops are an important concept when constructing meshes. They are so important that Blender provides a shortcut for selecting an entire edge loop with one click: ALT + RMB on an edge or vertex that is part of the loop you want to select, and it will select the entire loop. Alternatively, ALT + SHIFT + RMB adds an edge loop to the selection; or, if the part you click on is already selected, it will *deselect* the entire loop.

For example, try experimenting with a UV sphere: every line of "latitude" and "longitude" in this mesh is an edge loop.

2.3.6 Loop Cuts

Sometimes you need to add more vertices to the interior part of a mesh, perhaps to flesh in some detail. The loop cut function lets you add more edge loops between existing ones.

Ensure you are in Edit mode. It doesn't matter what parts of the mesh are currently selected. Press CTRL + R to activate the Loop Cut function. You will see a magenta-coloured loop wrap itself around different parts of the mesh as you move the mouse. You can press RMB or ESC to abandon the operation at this point, or once you see the loop appearing around the correct part of the mesh, you can use LMB or ENTER to proceed. Now the magenta colour changes to the usual orange-yellow selection highlight, and will now restrict itself to sliding along this section of the mesh as you move the mouse. If you press LMB or ENTER at this point, you will end up with a new loop of vertices and edges at the last-shown point, while RMB or ESC will still create the new loop, but leave it positioned at the midpoint.

When the loop is still at the magenta stage, you can use the mouse wheel to increase the number of cuts to 2 or more. You can also type a number of cuts using 0KEY ... 9KEY .

2.3.7 Edge Loop Deletion

Conversely, you can get rid of edge loops as well, reducing the complexity of the surface without leaving holes in it. Select the edge loop (the quick way is ALT + RMB on a component edge or vertex as described above), then bring up the deletion menu (DEL or X) and select "Edge Loop". The selected loop will disappear, and adjacent edges and faces will be merged.

2.3.8 Subdividing Parts

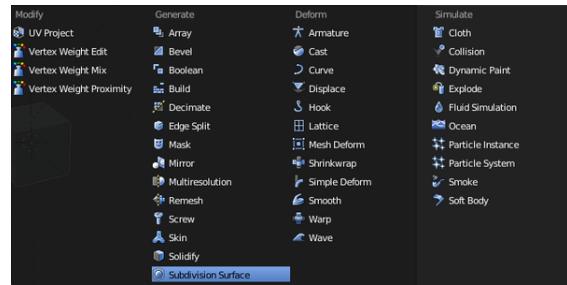
A loop cut always cuts a complete loop. Alternatively, you can subdivide just a selected part of the mesh: make your selection, then press W to bring up the Vertex Specials menu, and select the top option, "Subdivide". This will create one cut, but a panel will appear at the lower left of the Toolshelf (T to make it visible at the left of the 3D view if it's not), where you can alter the number of cuts and other settings. This same option is also available on Edge Specials CTRL + E .

Another option is the second one on the W menu: "Subdivide Smooth". This one computes a **Catmull-Clark** interpolation to give more of a curve rather than a flat subdivision.

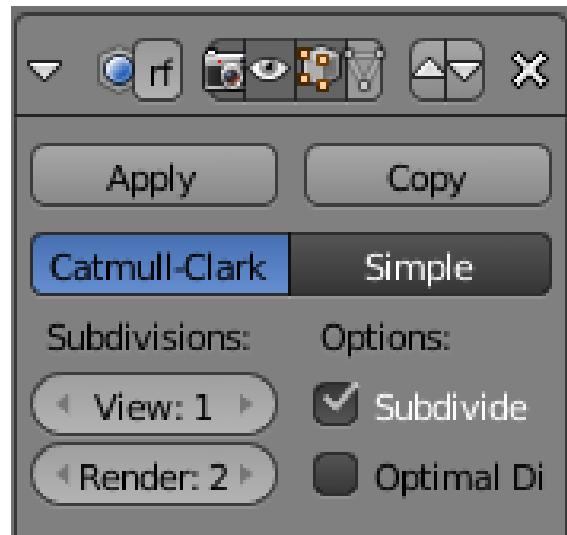
2.3.9 Subdivision Surface Modifier

A modifier causes some change to the geometry of an object just before it gets rendered. The change does not affect the object as you view and edit it in the 3D view, or as it is stored in the document (unless you apply the modifier, which makes the change permanent). This allows you to create some complicated effects at render time, while the original mesh stays simple and easy to edit. Modifiers for the active object are applied and controlled

in the Modifiers  tab in the Properties window.

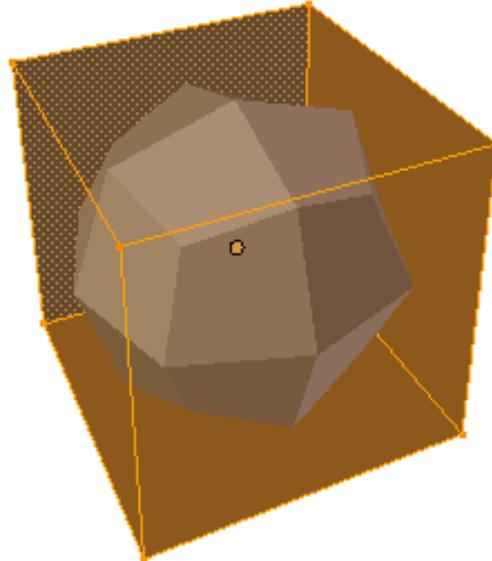
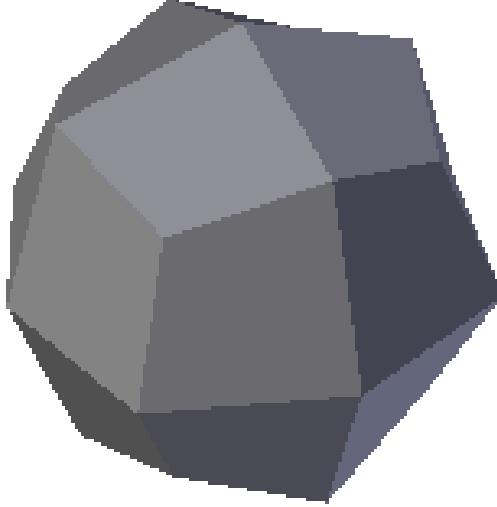


A subdivision surface modifier (also known as a "sub-surf" modifier) applies the Catmull-Clark interpolation discussed above as a modifier. Being a modifier, it applies to the entire object, not just to selected vertices. But since the original mesh is preserved, you use it as a *control cage* to adjust the shape of the interpolated curve.



Start with the default cube selected in Object mode, as usual. Go to the Modifiers tab in Properties. When you select "Subdivision Surface" from the "Add Modifier" menu, a new panel appears as at right. Notice the two value sliders under the "Subdivisions" heading; 'View' controls the level of subdivision within the 3D view, while 'Render' applies to the actual render; the higher the number of levels, the closer to a curve the interpolated geometry becomes. Having two separate settings for working

environment and render allows for faster operation in the 3D view, with the usual tradeoff of lower quality, while still allowing maximum quality for the final render.



As soon as you add the modifier, the appearance of the cube should change to look something like at right (here shown with just one level of subdivision).

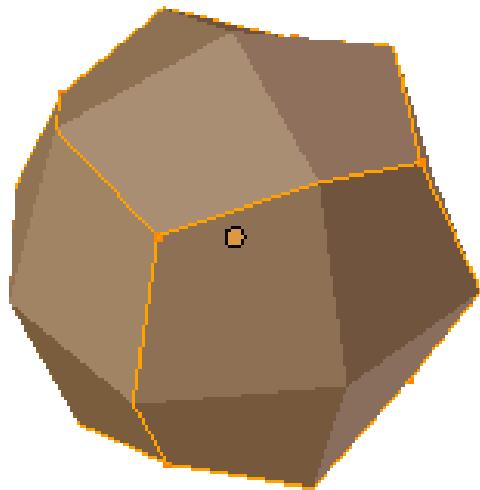
The upper part of the panel (from the “Apply” and “Copy” buttons upwards) is common to all modifiers. Note the X button at the right. Clicking it gets rid of the modifier. Notice also a group of 4 icon buttons in the middle, the leftmost two look like a camera and an eye. The icons are defined as follows (from left to right):

- Use the modifier during rendering
- Show the modifier effect in the 3D view
- Show the modifier effect in the 3D view in Edit mode (if this is unchecked and the previous one is checked, the modifier effect disappears while in Edit mode)
- Show the mesh as though the modifier were applied to it in Edit mode.

Unchecking the first one lets you disable the modifier without losing its settings. The remaining three can be handy if you’re trying to disentangle the effects of different modifiers during editing.

When the third button is enabled, the mesh will look like this in Edit mode. The original mesh remains highlightable and editable. A preview of the effect of the modifier is also visible, and responds immediately to any changes made to the original mesh. (Try moving some vertices around, and see what happens.)

The fourth button goes one step further and acts as if the modifier has already been applied, while allowing you to edit only those parts corresponding to the original mesh.



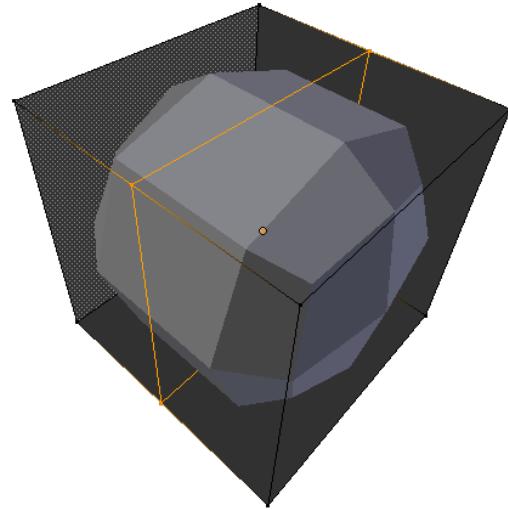
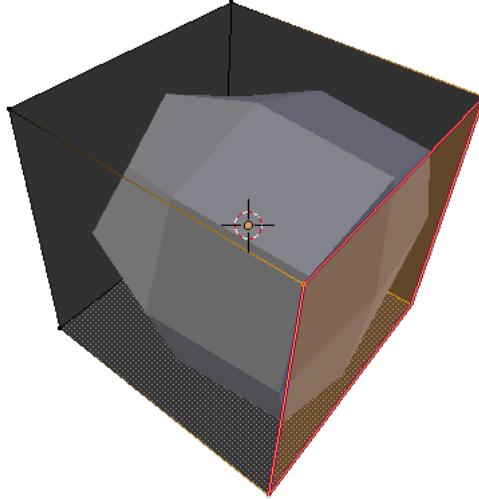
(This button affects the behavior of the third button. It cannot be used independently, and may disappear if the third button is unchecked.)

2.3.10 Sharpening the Curves

The subdivision surface modifier offers much more control over the resulting shape than might be apparent from above. For example, you may not want uniform curvature everywhere, you may want some parts of the shape to have sharper edges. This can be achieved in two ways:

- by applying a *crease* value to selected edges
- by strategic positioning of additional vertices in the control-cage mesh.

Applying a Crease

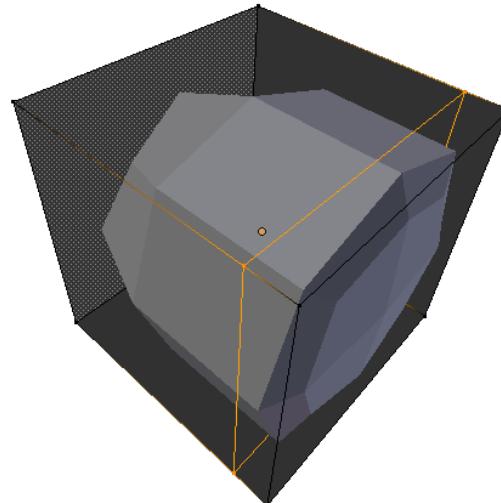
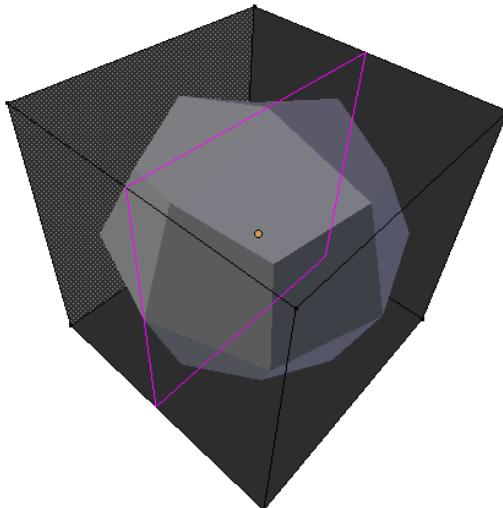


Press LMB or ENTER ...

Select the edges where you want the curve to be sharper. Press SHIFT + E . Note how the curve gets pulled more or less closer to those edges as you move the mouse. The selected edges take on a magenta colour, indicating they have a nonzero crease value applied.

The crease value can be seen and edited in the Transform panel at the top of the Properties Shelf at the side of the 3D view (you can toggle its visibility with N . Values can range from 0.0 (no crease, the default) to 1.0 (maximum sharpness of the edge).

Adding Vertices



... and move the mouse so the newly-added loop moves closer to one side of the cube. See how the subdivided mesh develops a sharper curve on this side?

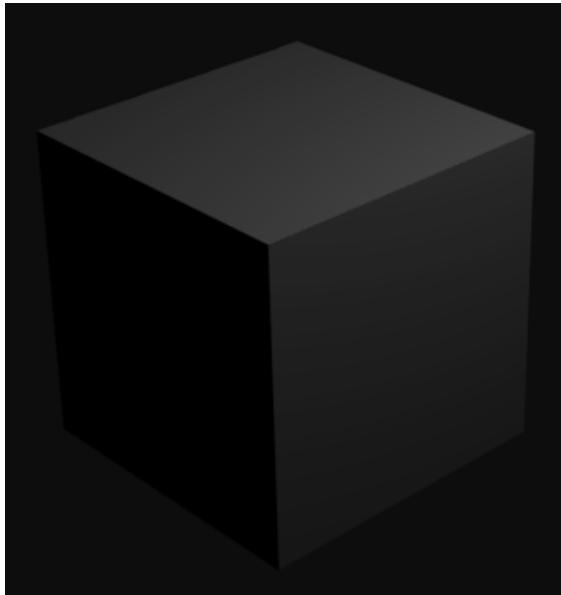
To confirm the placement of the new loop, press LMB or ENTER .

Which to Use?

For example, start with the subdivided cube example as above. Press CTRL + R to start a loop cut, and position the magenta outline something like this:

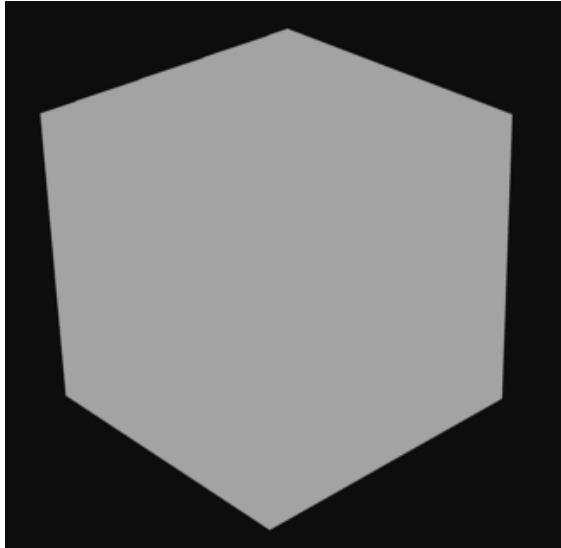
The basic principle is, the closer together the vertices are, the more control you have over the curve at that point. So the question is, do you just want a sharper edge, or do you want more detail? That will govern whether you need to add vertices, or just apply a crease to the existing edges.

2.4 Quickie Lighting



Open a new default Blender document. Without doing anything else, hit F12 to render the default cube with the default settings. The result should look something like the image to the right.

Note the lower left visible face of the cube is completely black because the default light is at the upper right.



Go back from the render to the 3D view (F11). Now select (with RMB) the default light, and either delete it or move it to another layer (with M). Go to your World

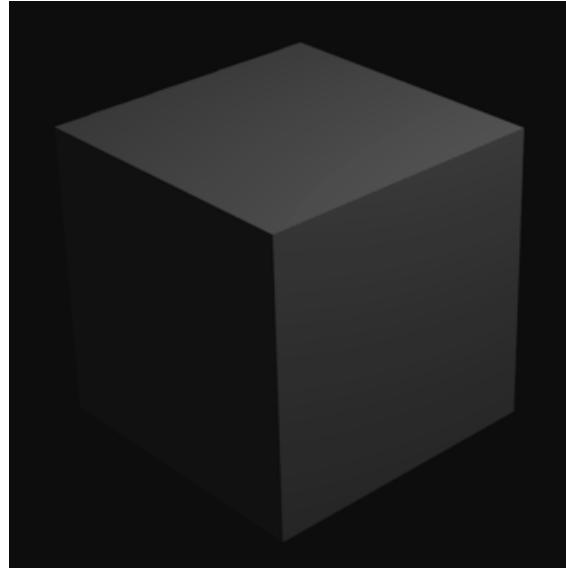
tab in the Properties window, and look for the Environment Lighting panel:

Check the box next to the title, and leave the “E:” (energy) value at its default 1.0. This gives us a pervasive, directionless light, illuminating all objects equally from



all directions, which means there will be no shadows. Do another render, and it should now look like the image to the right.

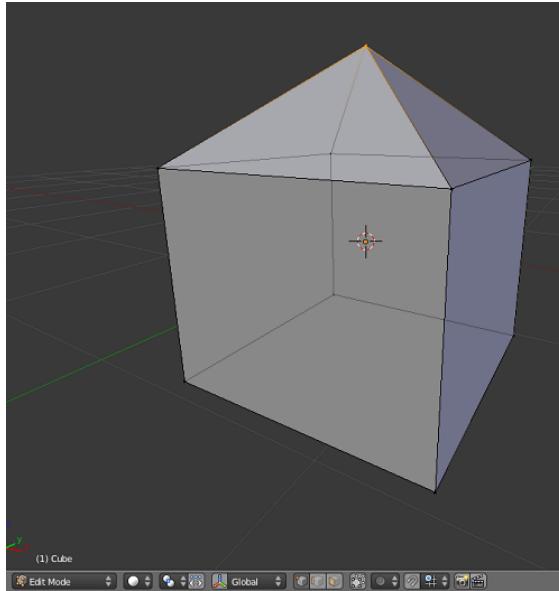
See how we have gone from inky-black shadows to no shadows at all. In the real world, lighting is almost never perfectly uniform, and this variation of light and shade is important to help us distinguish details of the scene around us. Without such variations, everything devolves into featureless blobs.



Now undo your deletion of the default lamp (or move it back to layer 1). Enable Environment Lighting again, but this time lower the Energy value to 0.1. Do another render, and it should now look like the image to the right. The shadowed face is still shadowed, but not enough to make it impossible to see any details it may have. This is usually the type of effect you want, unless you are aiming for really dramatic contrasts.

So the lesson is:

As you learn more, you will find that it is common to use two or three lights, or even more, to ensure proper illumination of a scene. In simple tutorials, where no explicit details are given about lighting, you can probably get by with the default light, plus some environment lighting (as we added earlier) to soften the shadows.



Your goal.

2.5 Quickie Model

In this module, you'll learn how to extrude and merge vertices of a mesh and how to save a model. This module also introduces the File Browser window type.

Your first model will be a house, which we will develop over the course of several modules. Here we will start with four walls and a pyramidal roof. Simple! Since you're going to use the default cube as a base, all you actually need to build is the roof!

Editing in Blender generally involves four steps:

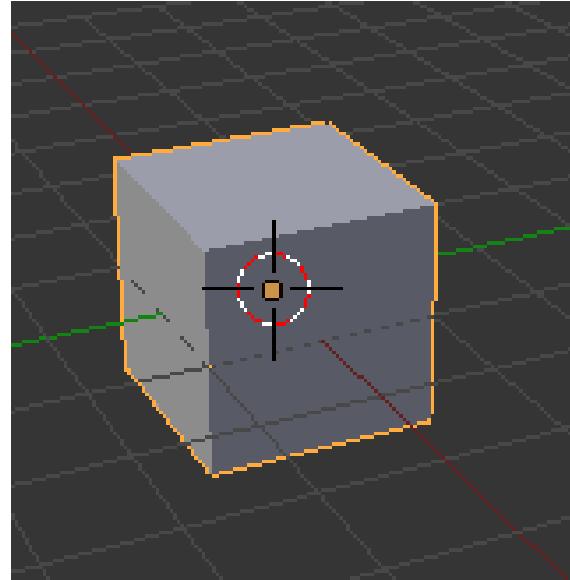
1. Selecting an object to edit.
2. Activating Edit Mode on that object.
3. Selecting part(s) of the object to act upon.
4. Specifying the action(s) to be performed on those parts.

2.5.1 Bring up the Default Cube

1. Launch Blender.
2. Load the factory settings using *File → Load Factory Settings*.

This should give you a perspective view of a scene containing three objects:

- a cube,
- a light source,
- a camera.



The default cube in Object Mode.

2.5.2 Setting up the Viewport

It will be easier to work on the roof of your house in a perspective side view:

1. Press Num3 to switch to a “perfect” right side view.

“Right Persp” will be shown on the top left of the 3D View. The “up” (Z) direction in the scene is now “up” on your monitor as well.

It will also help to zoom in a bit:

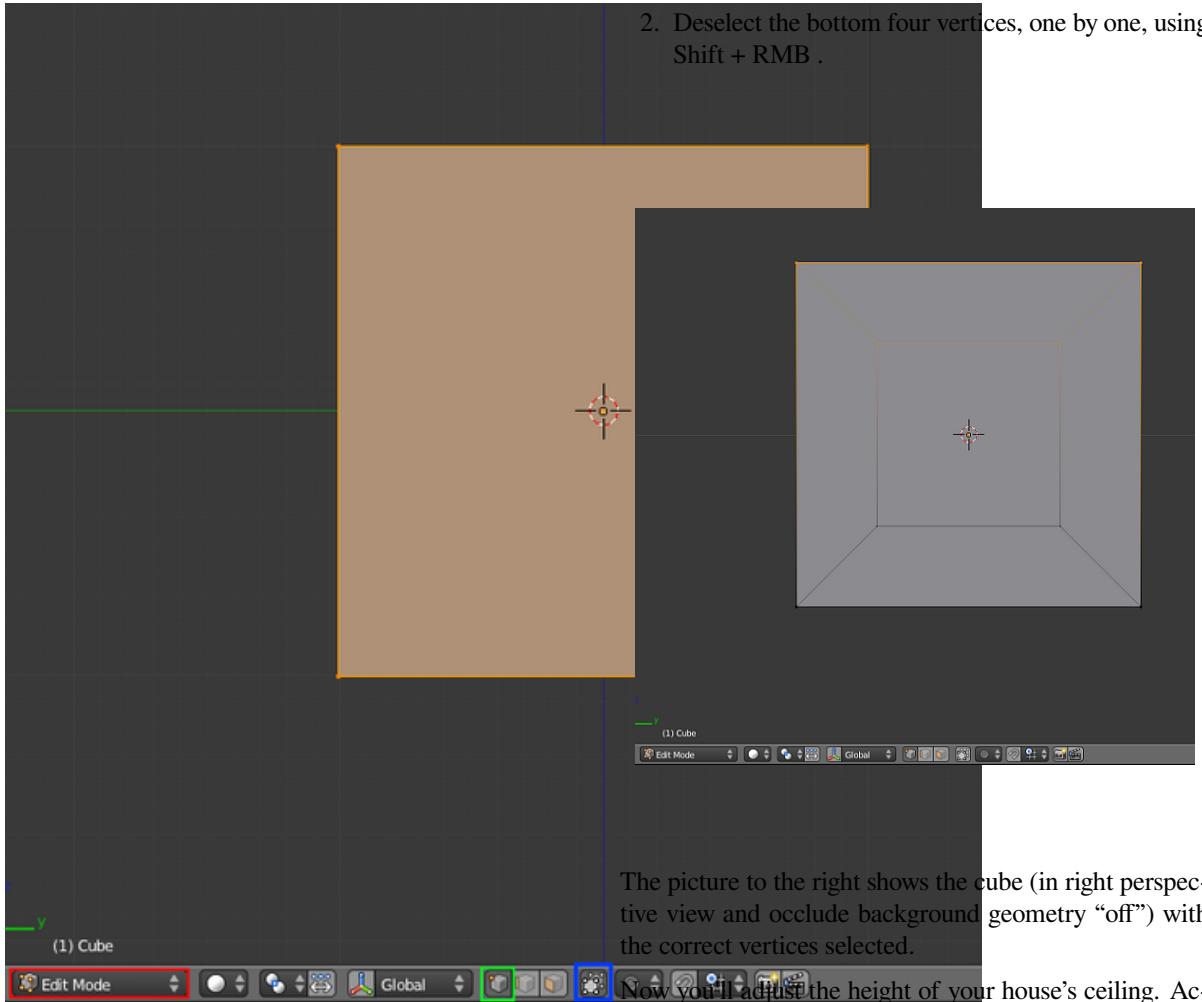
1. Make sure the 3D View window is active (which means your mouse cursor is in it).
2. SCROLL or press Num+ a few times until the cube is about 1/3 the height of the viewport.

Because you just loaded the factory defaults, the 3D transform manipulator will be enabled. For mesh editing, it will help to turn the manipulator off:

1. Make sure the 3D View window is active.
2. Press Ctrl + Space to toggle the manipulator on or off.

Press Tab once. This puts you into Edit Mode on the selected object, i.e. the cube.

Here's how the cube should look at this point:



The picture to the right shows the cube (in right perspective view and occlude background geometry “off”) with the correct vertices selected.

Examine the 3D View header to verify Blender is in Edit Mode and Vertex select mode with Occlude background geometry “on”.

The default cube is constructed as a mesh. Now that you're in Edit Mode, you can access the individual vertices, edges, and faces that make up the mesh. The default cube consists of eight vertices, twelve edges, and six faces.

2.5.3 Adjusting the Height

Right now, all eight vertices are selected, so any vertex edits you make will affect them all. For instance, if you were to move a vertex, the other seven vertices would follow. In order to build a roof peak for the house, you need to alter just the four top vertices of the cube. To do that, you must change the selection so that only *those* vertices are selected.

1. Turn 'Occlude Background Geometry' off, so you can see all vertices. Note, in newer versions, this button is called "Limit Selection to Visible." It is one of the buttons to the right of transform orientation which is to the right of the mode select which should be currently set to "edit mode".

2. Deselect the bottom four vertices, one by one, using Shift + RMB .

Now you'll adjust the height of your house's ceiling. Activate the **grab tool**:

1. Make sure Blender is in Edit Mode, with the relevant part(s) of the object selected.

2. Make sure the 3D View window is active.

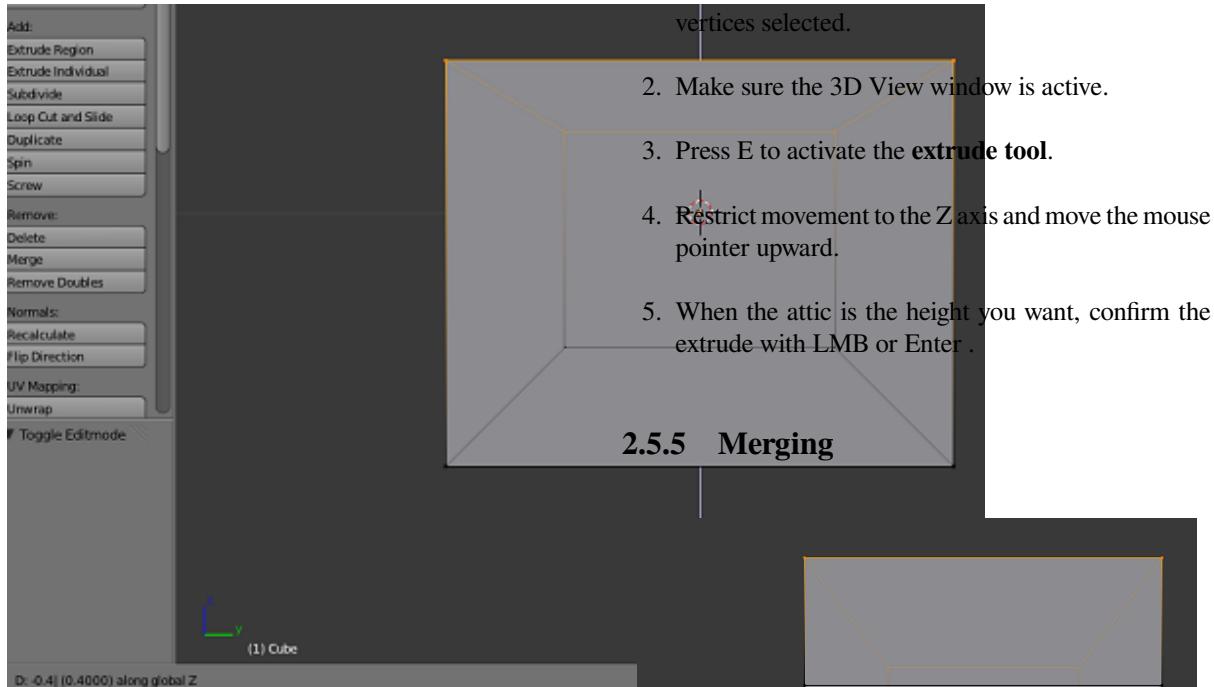
3. Press G .

The 3D View header will be replaced by numbers: “Dx: 0.0000 Dy: 0.0000 Dz: 0.0000 (0.0000)”.

You want to lower the ceiling without making the walls crooked. This is hard to do freehand, but happily the grab tool provides an option for doing just that.

With the grab tool activated:

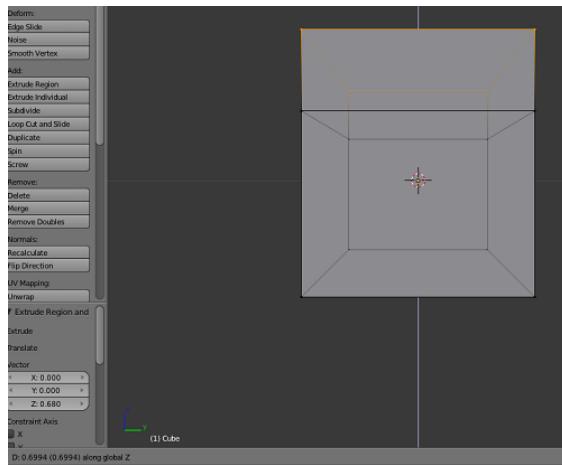
1. Press Z to limit movement to the global Z-axis.



Now when you move the mouse pointer around, Blender will adjust the height of your ceiling without making the walls crooked. (You can also limit grabs to the X and Y axes and in many other ways.)

When your ceiling is the height you want, confirm the grab with LMB (or Enter).

2.5.4 Extruding

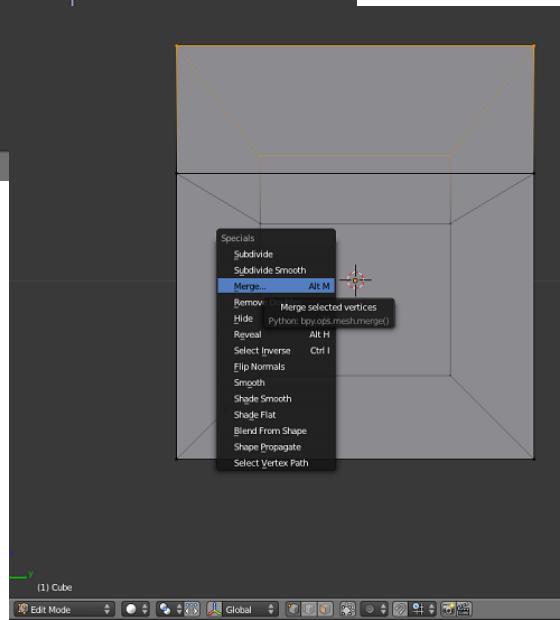


Step 7: the extruded attic, ready to confirm

Now you're going to “add on” to your house by **extruding**. Extrusion begins by duplicating selected parts of an object. Then the new parts are pulled away from the old ones, with new faces and edges created as necessary.

To add an attic to your house:

1. Make sure Blender is in Edit Mode, with the top four

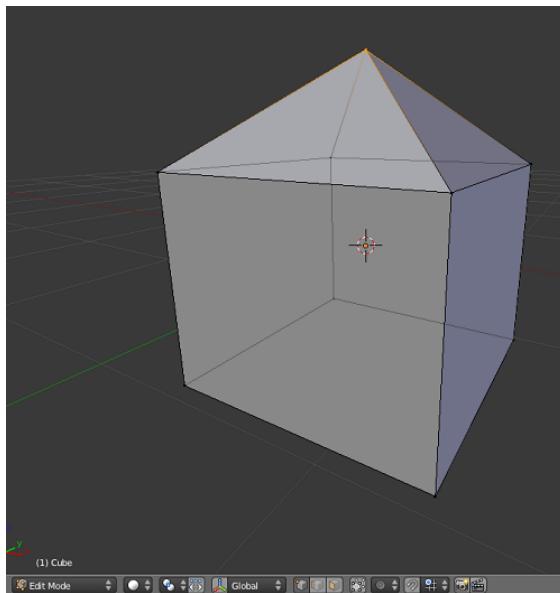


The Specials menu

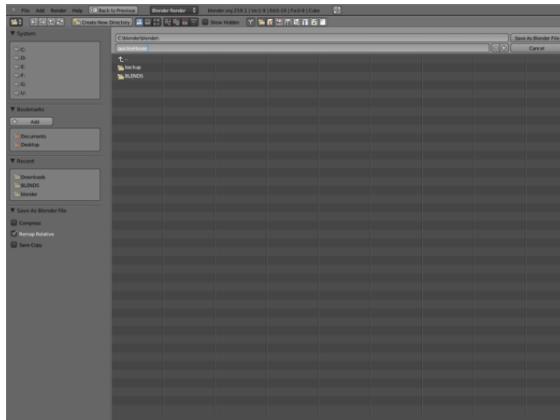
You can change the roof from a flat one to a pyramidal one by merging the vertices of the roof:

1. Make sure Blender is in Edit Mode, with the four top-most vertices selected.
2. Make sure the 3D View window is active.
3. Press W to bring up the *Specials* menu.
4. Select *Merge*. (You can also access this by pressing Alt + M .)
5. The *Merge* menu should pop up, select *At center*.

A message should appear on the Info header saying that 3 vertices have been deleted, this is because in order to merge four vertices into one, three vertices must be deleted.



Your house now has a pyramidal roof!



Saving the .blend

2.5.6 Saving your Work

We will be developing the house in later modules, so save your work now. To save the current scene in a .blend file:

1. Press F2 (or select *File* → *Save As*). The active window temporarily changes into a File Browser window.
2. Navigate to the directory (folder) where you want to write the file by clicking LMB on directory names in the File Browser window. (Clicking on “..” will take you up one level.)
3. If you wish to name the file something other than “untitled.blend”, type a filename in the text box to the left of the “Cancel” button. (The .blend suffix will be added automatically.)
4. Click LMB on the “Save File” button. As soon as the save operation is complete, the window will automatically revert to its former type.

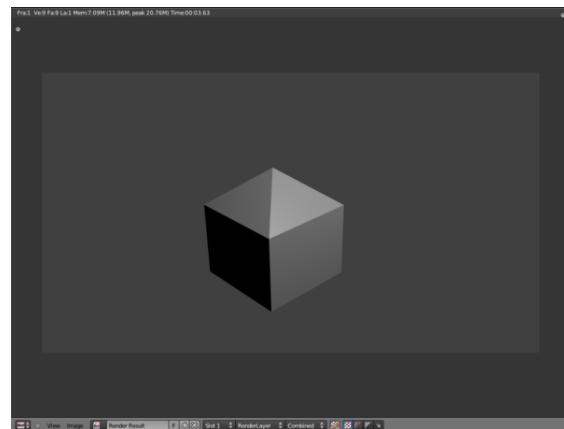
Saving Further Changes

Once you have saved your work to a file for the first time, you can save subsequent changes to the same file name by pressing **CTRL + S** and confirming you want to overwrite the existing file.

2.5.7 Additional Resources

- A Video Tutorial on Edit Mode

2.6 Quickie Render



If you haven't completed [the “Quickie Model” module](#), do so now. You will need the resulting model for this module.

Now that you've created your first model, you'll probably want to try rendering it. Your first render, with a single light source and only nine faces, should finish quickly. However, as your 3D scenes become more complex, you'll find that rendering can take a long time.

In this module, you'll render your quickie model and save the result in various file formats. You'll also learn how to aim cameras and create lamps.

2.6.1 Rendering the Quickie Model

1. Launch Blender and load factory settings.
2. To load the house model from the previous module, select *File* → *Open Recent*, and select the file you saved. Alternatively, press F1 or select *File* → *Open*, find the file, and open it. As soon as the operation is complete, the window will revert back to its former type.
3. Press F12 or select *Render* → *Render Image*. This opens the Image Editor so you can watch the render progress.

If F12 is in use by the window manager...

- With the new Apple keyboard, use Fn + F12 to avoid the Mac Dashboard.
- With Macintosh OS X 10.5, use Alt + Fn + F12 .
- With Gnome, use Alt + F12 to avoid the Gnome Search Dialog.

2.6.2 Seeing Your Render

By default, pressing F12 will switch to the UV/Image Editor window, and show your render there. You can switch back to the 3D view with F11 . Pressing F11 in the 3D view will switch you to the UV/Image Editor window without redoing the render, i.e. you will see the same image as last time.

2.6.3 Aiming the Camera

If you don't get a picture of the house, or if the picture is not framed well, try moving or re-aiming the camera:

1. Press Esc to get back to Edit Mode, if needed.
2. Press Num0 to take the camera's viewpoint.
3. Press Shift + F to put the 3D View window into camera fly mode.

In camera fly mode, you can:

- Pan and tilt by moving the mouse pointer up, down, left, or right.
- Accelerate by SCROLL forwards.
- Decelerate by SCROLL backwards.
- Press any key or button to exit fly mode.

(It works differently in version 2.70 and later, more like a FPS game with possibility to slide and so on, buttons are regular FPS controls)

When you're done positioning the camera, try rendering again.

2.6.4 Lighting

If your cube is completely black, you may not have a lamp in the scene. Either the default lamp got deleted, or you're using a version of Blender that doesn't provide a default lamp.

To add a lamp:

1. Make sure Blender is in Object Mode.
2. Place the 3D cursor where you want the lamp to go; or add the lamp then immediately grab it, and move it somewhere else.
3. Press Shift + A .
4. In the popup menu, select *Lamp → Point*.

2.6.5 Saving the Render

Saving the scene (with F2 , for instance) does not save any renders. Saving renders is a separate step.

To save your current render:

1. Make sure you are in the Image Editor. If not press F12 to render
2. Press F3 . This temporarily changes the active window into a File Browser window.
3. Navigate to the directory (folder) where you want to write the file.
4. Type a filename in the text box (to the left of the “Cancel” button).
5. To the left of the window, choose your preferred file type.
6. Click LMB on the “Save as Image” button. As soon as the save operation is complete, the window will revert back to its former type.

2.6.6 Renderer Selection



Blender offers a choice of different rendering engines for producing images. The menu for selecting from these appears in the Info window (the thin one that contains the menu bar at the top of the default layout). In most of

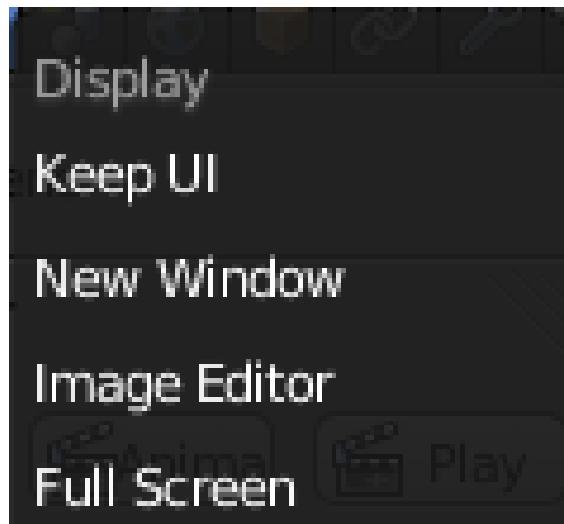
these tutorials, you will leave this choice set at Blender Render. But it is worth knowing what other choices are available:

- Blender Render—the oldest renderer, commonly known as the Blender Internal renderer. Built into Blender right from its early days. Can still produce good results with the right tricks, but considered by the Blender developers to be antiquated and not worthy of continuing development.
- Blender Game—this is the renderer used by the Blender Game Engine. Designed to be fast enough for interactive use in a game, which means there are limitations in the quality of renders it produces. You also use this renderer to create rigid-body physics simulations.
- Cycles Render—for this and other choices, see Advanced Rendering.

2.6.7 Render Control



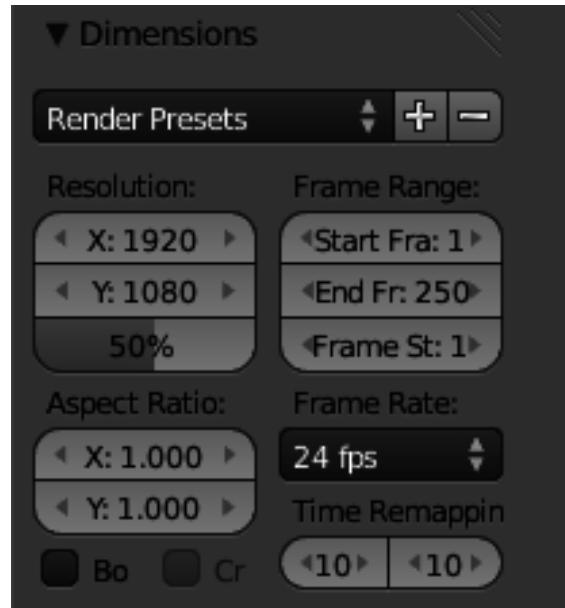
The top panel under the Render tab in the Properties window shows 3 buttons and a menu. The first button renders a single frame, equivalent to F12. The other two buttons are more relevant to **animations**.



The “Display.” menu controls what happens when you press F12 : the default “Image Editor” causes the 3D

view to be switched to the UV/Image Editor showing the rendered image. “Full Screen” causes the UV/Image Editor display to take over the entire screen, while “New Window” makes it appear in a separate OS/GUI window (similar to how older versions of Blender used to work). Finally, “Keep UI” causes no changes to your window layout at all; you have to explicitly bring up the Image Editor with F11 to see the rendered image.

2.6.8 Render Image Dimensions



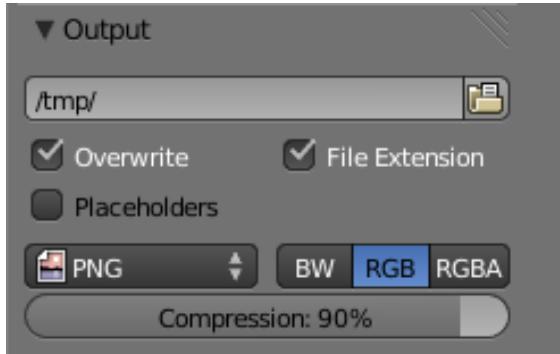
You can control the size of the image that Blender creates when rendering. This is specified in the “Dimensions” panel under Render properties. Apart from the menu at the top, the settings in this panel are grouped into two columns:

- The column on the left controls settings for a single image.
- The column on the right specifies additional settings for rendering a whole sequence of images as part of an animation. These settings will be discussed later.

At the upper left, under “Resolution:”, we have the dimensions in pixels of the image (the default settings are 1920×1080 as shown in the screenshot), plus an additional scale factor slider below (showing 50% by default). With these settings, the image will actually be rendered at $(1920 \times 50\%) \times (1080 \times 50\%) = 960 \times 540$. Having the scale factor is a convenience. Rendering smaller, lower-quality images is faster, which speeds up initial work on your model, but you’ll want full quality for the final result. Instead of mentally having to work out numbers for render quality, you can simply set the resolution to full quality,

and use the scale factor to reduce this to, say, 50% or 25% for interim work, then set it to 100% for the final output.

2.6.9 Image File Formats



You set the format and location for saving rendered images in the “Output” panel under the Render properties.

In current versions of Blender, the default format for saving rendered images is **PNG**. This is a *lossless* format which has the option for *alpha transparency* (which means the sky background is replaced by transparent pixels—enabled by clicking the “RGBA” button). This is a good format if you intend to do further work with the image (e.g. in an image editor like Gimp or Photoshop), but the files can be large.

JPEG is a *lossy* image format, which means it throws away information that the human eye doesn’t see. This produces much smaller files than PNG, and is adequate if you just want to upload the render directly for use in a Web page or other such document, but is not the best choice if you intend to do further processing of the image. It also doesn’t support alpha transparency.

To change the render file format:

1. Switch to the Render tab in the Properties window.
2. Look for the “Output” panel.
3. Click LMB on the popout menu with the current file format.
4. Select your preferred format.

2.6.10 Additional Resources

- the “Output Formats” module
- Tutorial on Using Multiple Cameras ← Pictures are missing from this tutorial
- Ira Krakow’s Basic Blender Camera Positioning (Rigging)

2.7 Enter the World

The World settings control the background or sky settings for your scene. A scene doesn’t have to have a World, in which case, the background will simply be black. If you save your images with a transparent Sky setting, the background doesn’t matter. But in other situations, you will want to control what appears here.



Here is what the top two panels in the World Context look like in a new default document. (Settings further down for Ambient Occlusion, Environment Lighting and Indirect Lighting will be discussed later, when we discuss lighting.)

“Ambient Color” is a sourceless, shadowless light, coming from all directions, applied to all objects. Trouble is, its effect is very “flat”, i.e. it washes out detail. It’s probably better to use the Environment Lighting or Ambient Occlusion options (in panels further down) to soften murky shadows.

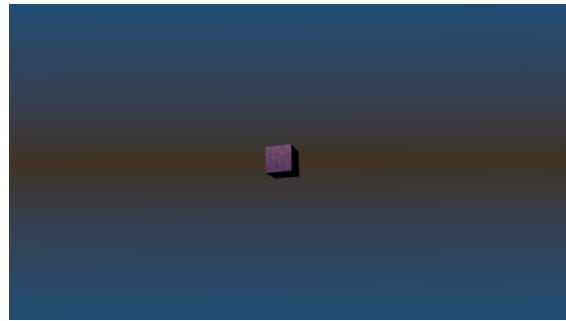
With none of the boxes checked, “Zenith Color” has no effect, only “Horizon Color” does. The sky will simply be a flat expanse of this colour. Selecting “Paper Sky” and “Real Sky” on their own has no effect, they work only in conjunction with “Blend Sky”.

Check “Blend Sky” on its own. Now the sky takes on a gradient from “Zenith Color” at the zenith (straight up) to “Horizon Color”, not at the horizon, but at the nadir (straight down).

Check both “Blend Sky” and “Real Sky”. Now you get “Horizon Color” at the horizon, with a gradient to “Zenith Color” at both zenith *and* nadir.

The effect of “Blend Sky” with “Paper Sky” is a bit more subtle. It means the horizon is always in the middle of the image, regardless of the orientation of the camera. The effect is more noticeable if you check “Real Sky” as well,

otherwise it looks little different from “Blend Sky” on its own.



it is on the horizon, even though the camera angle is the same as the previous image.

You can also add a texture to your sky. That will be discussed later.

2.8 Understanding the Camera

2.8.1 Real-World Cameras

Before discussing the camera in Blender, it helps to understand something about how cameras work in real life. We have become accustomed to so many of their quirks and limitations when looking at real photographs, that 3D software like Blender often expends a lot of effort to mimic those quirks.

When taking a photo with a real camera, a number of important factors come into play:

- the *focus* — because of the way lenses work, only objects within a certain distance range from the camera (the *depth of field*) will appear sharp in the image. Objects outside this range will begin to appear noticeably blurred, the blur getting worse the farther they are outside the focus range. The narrower the range of in-focus distances (the shallower the depth of field), the more quickly this blurring happens with objects outside it.
- the *exposure time* — how long the shutter remains open. The longer this is, the more light is captured, but also the more likely the image is to pick up *motion blur* from moving objects.
- the *aperture* — how wide the iris opening is. This is expressed, not as an actual distance measurement, but as a fraction of the *focal length* of the lens (loosely, distance between the lens and the image-capturing surface when the image is properly focused), written as f : thus, say, $f/2.8$ (“ f over 2.8”, not “ f 2.8”) is a larger number, hence representing a wider aperture, than $f/8$. A wider aperture increases the amount of light being captured *without* contributing to motion blur, but it reduces the depth of field. The extreme case of a *pinhole camera* has

Here is an example with contrasting horizon and zenith colours. I also set the camera field of view to 90°.



With only “Blend Sky” checked, a render looks like this.



With “Blend Sky” and “Real Sky” checked, this is how the render comes out. Note how the cube is noticeably off-centre relative to the horizon band, because the camera view is at an angle to the horizontal.

With all three of “Blend Sky”, “Real Sky” and “Paper Sky” checked, the result is this. The cube now looks like

a very tiny aperture with infinite depth of field (no need to focus at all), but captures very little light, so it needs a very well-lit scene, a long exposure, or a very sensitive image-capturing surface.

- the *sensitivity* of the image-capturing surface to light. In the days of film cameras, we talked about film sensitivity (“fast” film being more sensitive to light than “slow” film). Nowadays, with digital cameras we talk about the *gain* of the light-amplification system. High-sensitivity film was more likely to produce a grainy image. In a somewhat similar manner, high light-amplification in a digital camera is more likely to produce a noisy-looking image under low-light conditions.
- the *field of view* — how much of the scene the camera can see at once. A *wide-angle* lens gives a wider field of view, but you have to be closer to objects to be able to see them, and there is greater perspective distortion. At the other extreme, a *telephoto* lens gives a very narrow field of view, but can take pictures of things from much further away. A wide-angle lens also has a shorter focal length than a narrow-angle one (remember that aperture is expressed as a ratio of the focal length f), therefore the telephoto lens is going to capture *less* light than the wide-angle one with the *same* aperture width. You may also have heard of the *zoom lens*, i.e. one with a variable focal length. It can be adjusted from a wide-angle mode to a telephoto mode.

As you can see, many of these different factors interact with each other. The brightness of the image can be affected by the exposure time, the aperture, the gain sensitivity and the focal length of the lens. Each of these have side-effects on the image in other ways.

Blender and other computer graphics software are, in principle, free of the problems of focus, exposure time, aperture, sensitivity and focal length. Nevertheless, it is common to want to introduce deliberate motion blur into an image, to give the impression of movement. Sometimes it is useful to introduce a deliberately shallow depth of field, blurring objects in the background in order to draw emphasis to the important part of the image, i.e that which is in focus.

Exposure (the total amount of light captured in the image) is also less of a problem in computer graphics than in real-world photography, because in computer graphics you always have total control over the amount and placement of lighting in the scene. Nevertheless, if you’re not careful, you can produce overexposed (bright parts losing detail by saturating to a solid, featureless white) or underexposed images (dark parts losing detail by becoming solid black).

The field of view issue arises from basic principles of geometry, and Blender’s camera is just as much subject to that as real cameras.

2.8.2 The Camera In Blender

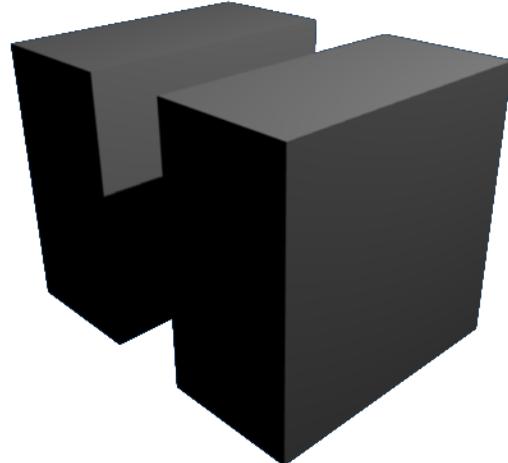
Here we are going to concentrate on the important issue of *field of view*.

You can change the field of view in two ways - move the camera closer to or farther from the scene (called *dollying* in film/TV production parlance), or change the angle of the lens (*zooming*). You do the latter in the Object Data  tab in the Properties window (the Camera has to be selected by RMB in Object Mode, or the required tab will not be visible).

Perspective is the phenomenon where objects that are farther away from the viewer look smaller than those nearby. More than that, different parts of the *same* object may be at different distances from the eye, leading to a change in the apparent shape of the object called *perspective distortion*. The mathematical theory of perspective was worked out by [Alhazen](#) in the 11th century, and famously adopted by the Italian Renaissance painters four hundred years later.

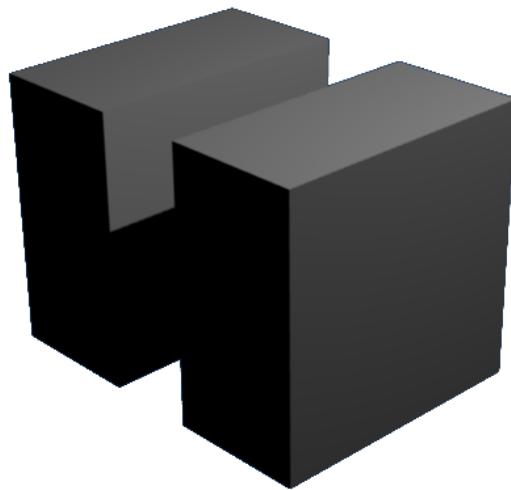
Here are two renders of the same scene with two different cameras, to illustrate the difference.

This one moves the camera closer but gives it a wider field of view:



This one moves the camera back, while narrowing its field of view, to try to give the scene the same overall size.

The latter is like using a “telephoto” lens with a real camera. Notice how the wider field of view gives you a greater perspective effect. The boxes are all *cuboids*, with parallel pairs of opposite faces joined by parallel edges, yet there is a noticeable angle between notionally-parallel edges in both images, which is more pronounced in the upper image. *That* is what perspective distortion is all about.



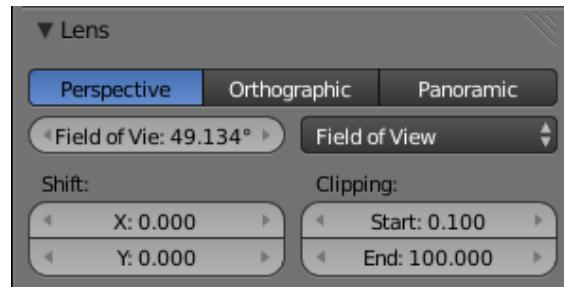
Specifying the Field of View

When you select a camera object, its settings become visible in the Camera Context in the Properties window, which should initially look something like at right.

Photographers are accustomed to working in terms of the focal length of the lens - longer means narrower field of view, shorter means wider field of view. But the field of view also depends on the size of the sensor (image capture area). Modern digital cameras typically have a smaller sensor size than the exposed film area in older 35mm film cameras. Thus, the focal length measurements have to be adjusted accordingly, in order to give the same field of view.

Blender allows you to work this way, by specifying the focal length in the “Lens” panel, and the sensor size in the “Camera” panel. It even offers a “Camera Presets” menu, which sets the sensor size for any of a range of well-known cameras.

Also, you might be doing compositing of your computer-generated imagery on top of an actual photograph. In which case, to make the results look realistic, you need to closely match the characteristics of the camera and lens that were used to take the photo. If you know the lens focal length and camera sensor size, it makes sense to be able to plug those values in directly.



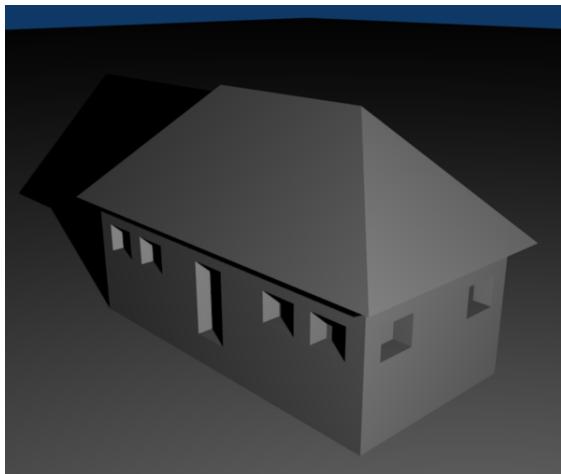
But if you’re *not* doing photo compositing, but generating completely synthetic imagery, you might consider this a somewhat roundabout way of working. Why not specify the field of view directly as an angle?

Blender allows for this as well. From the popup menu in the Lens panel that says “Millimeters”, select “Field of View” instead, and the Focal Length field will turn into a Field of View field, showing the angle in degrees directly. This is *much* easier to relate to the geometry of the scene!

2.8.3 See Also

- Improving Blender Renders with Photography Techniques — another explanation covering similar ground.

2.9 Improving Your House



Your goal.

In this module, you'll refine the house model you created two modules ago. In the process, you'll learn how to access Blender's predefined meshes and how to set a pivot point. You'll also learn how to select, extrude, delete, and subdivide the edges and faces of a mesh model.

To begin, set up Blender as follows:

1. Launch Blender and load the factory settings.
2. If you have a numpad, make sure NumLock is on.
3. Load the house model you created in the “Quickie Model” module.
4. If the 3D manipulator is active, disable it.
5. Adjust the viewpoint until you can clearly see two walls of the house and two sides of the roof.

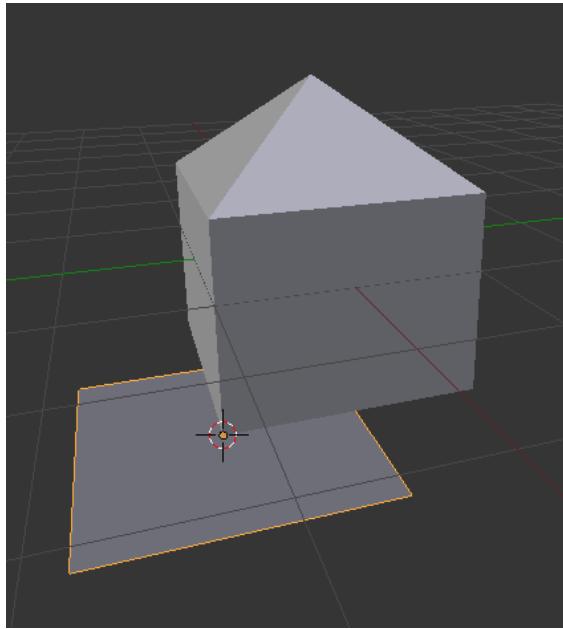
2.9.1 Adding a Ground Plane

Your house needs some ground to rest on. You can model the ground as an object in your scene. Blender has many predefined mesh objects built in. Happily, one of these is a flat, square surface.

Recall that new objects are added at the 3D cursor. Before creating the ground, you should position the cursor at ground level:

1. Select the house by clicking RMB on it.
2. Enter Edit Mode by pressing Tab .
3. Select one of the bottom vertices by clicking RMB on it.
4. Bring up the *Snap* menu by pressing Shift + S .
5. Choose *Cursor to Selected*.

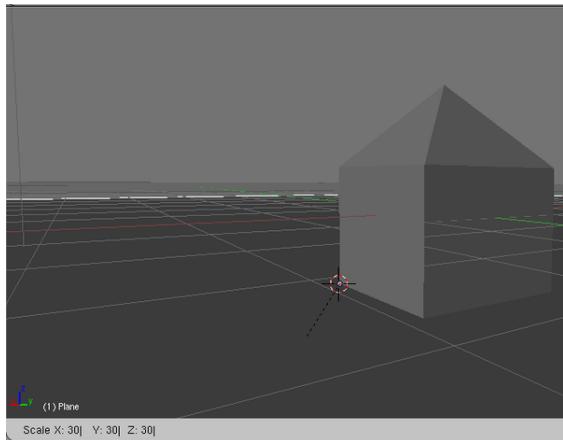
6. Leave Edit Mode by pressing Tab so your ground is created as a separate object.



The ground added.

Now create the ground object:

1. Activate a 3D View window.
2. Press Shift + A .
3. Choose *Mesh → Plane*.



The scaled ground.

To enlarge (or scale) the ground object, use the scale tool:

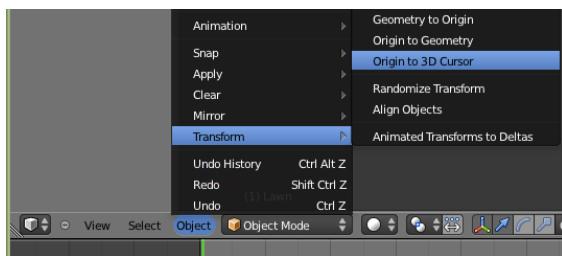
1. Make sure Blender is in Object Mode.
2. Select the ground by clicking RMB on it.
3. Activate the scale tool by pressing S .

4. Type 7key to enlarge the ground 7x.
5. Press Enter or LMB to confirm and exit the scale tool.

2.9.2 Scaling with a Pivot

Suppose you want to shrink the house by 50%. As you can probably guess, this would be done with the scaling tool. However, if you did so right now without the right pivot point, the reduced house would no longer rest on the ground. Blender scales (and rotates) objects around a pivot point, which by default is located at the median point (geometric center) of the selected object(s).

In order to scale the house while keeping its base on the ground, you need the pivot point to be at ground level. Since the 3D cursor is at ground level, you can do this as follows:



Origin to 3D Cursor Menu Item.

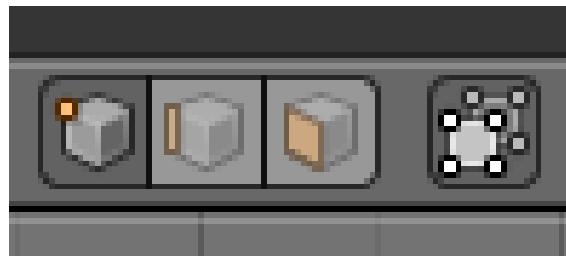
1. Make sure Blender is in Object Mode.
2. Select the house by clicking RMB on it.
3. In the 3D View header, click LMB on menu item “Object” and put the mouse cursor over *Transform* and select *Origin to 3D Cursor* from the pop-up menu. This can also be done with Ctrl+Shift+Alt+C, select *Origin to 3D Cursor* from the pop-up menu. (A.K.A. select “Object” which is just left of where you’ve been going into object mode/edit mode, as shown in the image. So Object>Transform>Origin to 3D Cursor)

The origin of the house is now at the center of the 3D cursor. If you scale the house, the place where the 3D cursor is located will remain fixed, and everything else will expand or contract from that point. The pivot is marked with an orange-filled circle. Do not mistake it for a selected vertex.

2.9.3 Edge Selection

It is often useful to select edges instead of vertices.

1. Make sure Blender is in Object Mode.



The select mode buttons.

2. Select the house by clicking RMB on it.
3. Press Tab to enter edit mode.
4. Click LMB on the Edge select mode button in the 3D View header.

In Edge select mode, edges appear as orange or white line segments when they’re selected and as black line segments when they’re not.

Just as you selected vertices in Vertex select mode, you can now select (and deselect) edges in the same way as vertices. This is also the same for Face select mode.

If you have difficulty selecting particular edges with the mouse...

It may be because those edges are doubled. This can happen if you cancel an extrude operation and forget to undo the duplication. Here’s a solution:

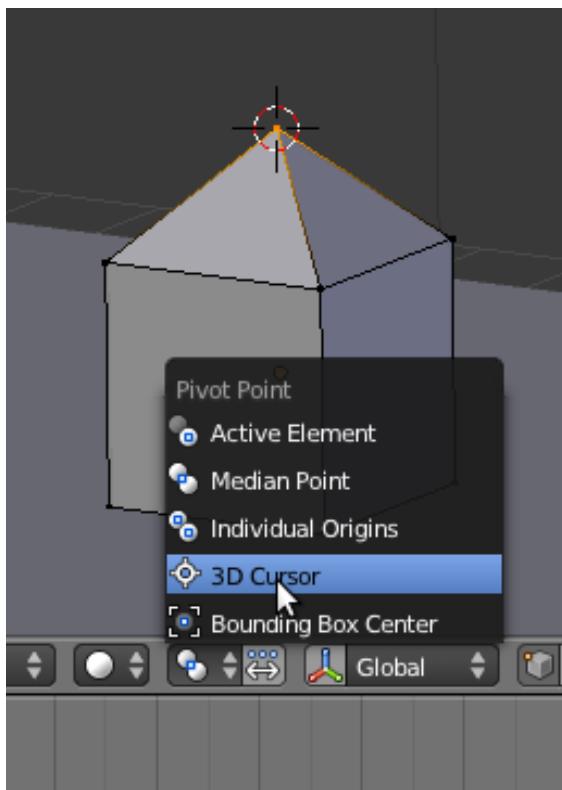
1. Switch to Vertex select mode.
2. Activate a 3D View window.
3. Select all vertices by pressing A once or twice.
4. Press W to bring up the “Specials” menu.
5. Choose *Remove Doubles*.

2.9.4 Extruding Edges

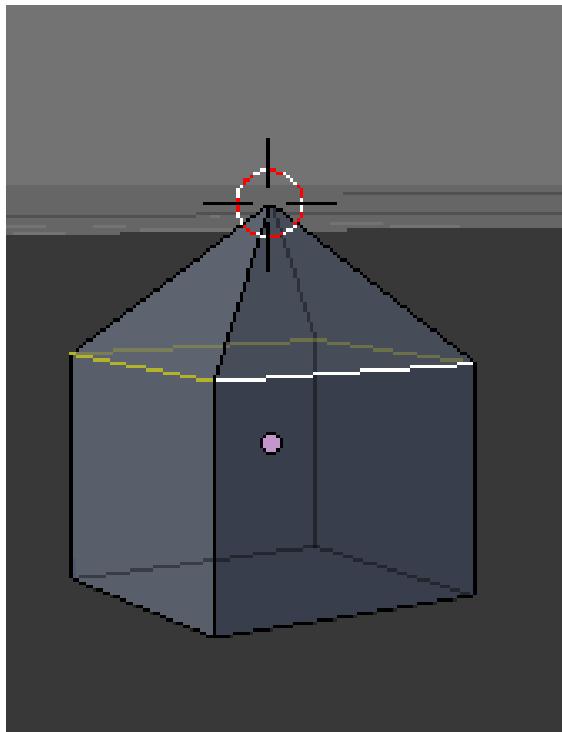
You can extrude edges in much the same way as you extrude vertices.

To add an overhang to the roof of your house, first move the pivot point to the peak of the roof:

1. Switch to Vertex select mode.
2. Select just the vertex at the peak of the roof.
3. Press Shift + S to bring up the Snap menu.
4. In the Snap menu, choose *Cursor to Selected* to move the 3D cursor to the peak.



Step 5

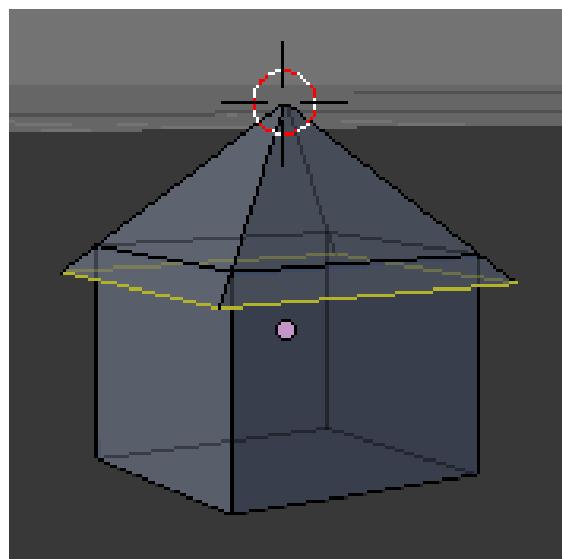


After step 2

5. Use the “Pivot” menu (located to the left of the 3D Manipulator button) in the 3D View header to change the pivot to “3D Cursor”.

Now extrude by scaling from that point:

1. Switch to Edge select mode.
2. Select just the four edges at the base of the roof.
3. Press E to activate the extrude tool.
4. Press S to extrude by scaling uniformly from the pivot point.
5. As you move the mouse pointer away from the pivot point, the roof of your house will expand.
6. When the roof is the size you want, confirm by LMB (or pressing Enter).
7. Press CTRL + SPACE to toggle the manipulator on then make the overhangs slanted by holding LMB on the blue arrow that appears in the center of the house, and dragging down.



After step 7

2.9.5 Face Selection

It is often useful to select faces.



The select mode buttons after step 2

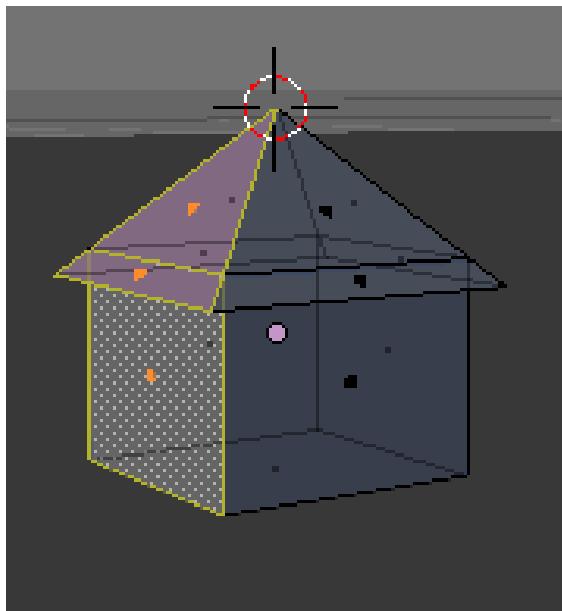
1. Make sure you're in Edit Mode on the house.

- Click LMB on the Face select mode button in the 3D View header.

In Face select mode, the center of each face is marked with a small square. Faces appear as orange or stippled grey areas with orange edges when they're selected (depending on which face is active), and as grey areas when they're not.

Just as you selected edges in Edge select mode, you can now select (and deselect) faces:

- If any faces are selected, press A to deselect all faces.
- If no faces are selected, press A to select all faces.
- To select a single face (and deselect the rest), click RMB (or Cmd + LMB) on the center of the face.
- To toggle the selection status of a face (without affecting the rest), click Shift + RMB on the center of the face.



The three faces on the +X side, selected

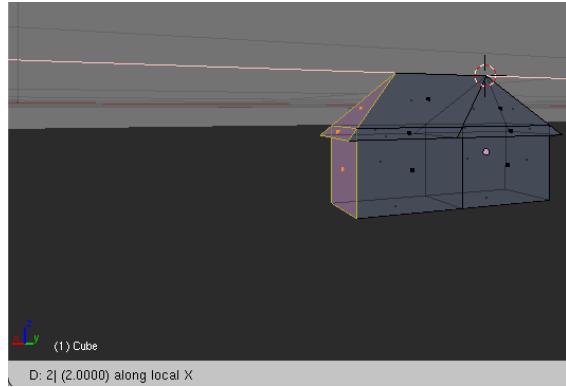
Use these techniques to select all three faces (two roof and one wall) on the +X side of your house, as shown.

- This reader would like to remind others that the positive direction of the axis is the direction the arrows point.

2.9.6 Extruding Faces

Just as you extruded edges to grow the roof, you can extrude faces to grow the entire house.

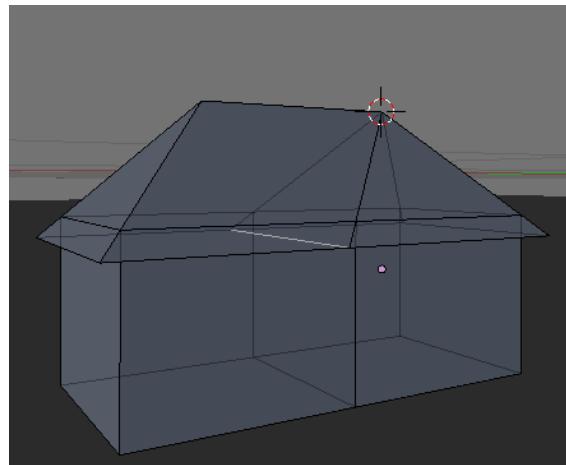
To double the size of your house without changing the pitch of the roof:



After step 6

- With the three faces on the +X side selected, activate a 3D View window.
- Press E to activate the extrude tool.
- Press X to extrude along the X axis
- As you move the mouse pointer in the +X direction, the +X half of your house will expand.
- Press 2 to expand by exactly 2 Blender units. (If you scaled your house earlier, you must change this value accordingly, e.g. scaling by 50% means you press 1 .)
- Confirm and exit the extrude tool by clicking LMB (or pressing Enter).

2.9.7 Deleting Edges



After step 2

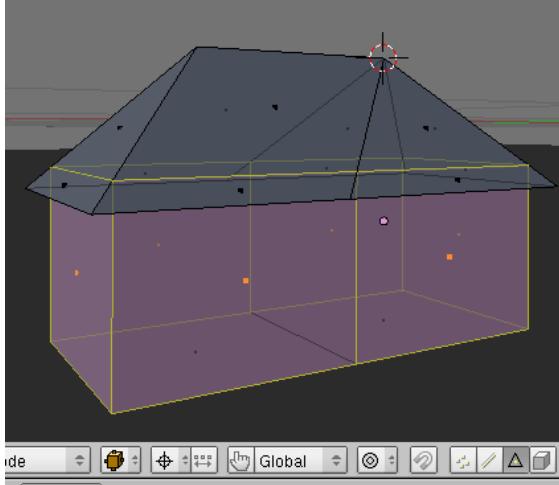
If you look closely at the model, you'll notice an extra edge connecting the seams between the two halves of the roof. To delete this edge:

- Edit the house object in Edge select mode.

2. Select just the edge you want to delete.
3. Press X or Delete .
4. When the “Delete” menu comes up, choose *Edges*.

2.9.8 Subdividing Faces

In order to add openings such as doors or windows to the walls of your house, you'll need to subdivide the wall (vertical) faces into smaller faces.



After step 2

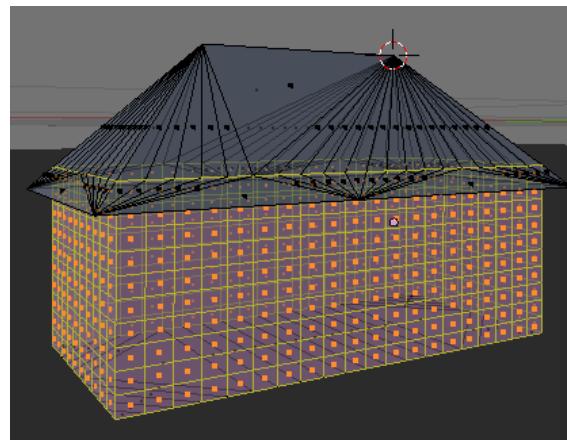
To subdivide each wall face into a 10x20 grid:

1. Make sure you are not in wire-frame mode (otherwise the occlude hidden geometry button will not appear)
2. Edit the house object in Face select mode.
3. Select all six wall faces of your house.
4. Press W to bring up the *Specials* menu.
5. Choose *Subdivide*.
6. Set the number of cuts to 9 in the Operator panel (also accessible through F6).

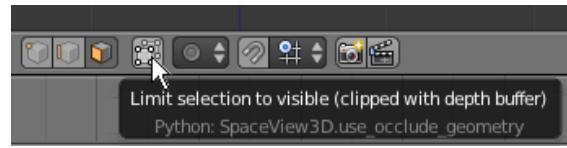
You might be wondering why to make 9 cuts instead of 10, the reason is that in case of dividing a finite surface along one axis there will be always n-1 cuts to generate n single faces. Here the number of cuts is applied in 2 dimensions. So, if you count the number of faces on the subdivided walls, you will find a 10x20 grid. The reason why there are 20 faces instead of 10 lengthwise is because you doubled the size of the house along the X axis (lengthwise).

Now you can extrude windows and doors:

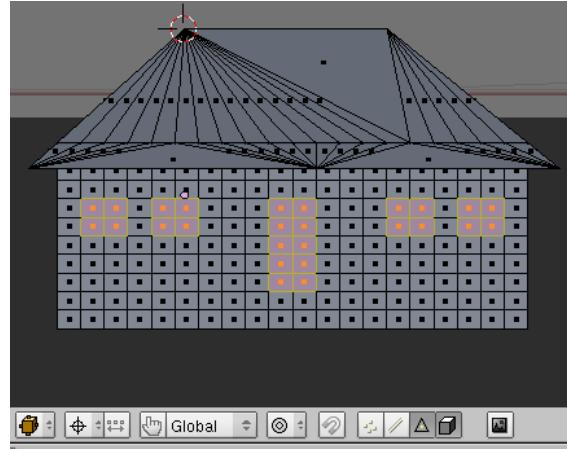
1. Edit the house object in Face select mode.



After step 6



Step 2



After step 3.2

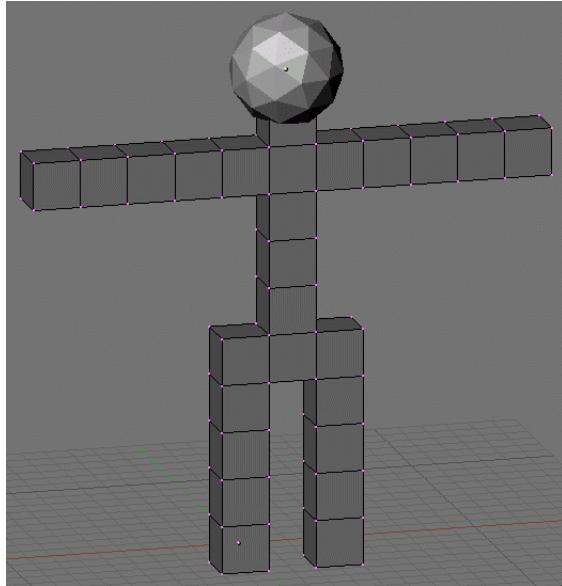
2. Turn on the “Limit selection to visible (clipped with depth buffer)” (for old Blender versions “Occlude background geometry”) option by clicking LMB on the toggle button in the 3D View header.
3. For each wall of the house:
 - (a) Go to the perfect view for that wall:
 - Num1 for “front”
 - Ctrl + Num1 for “back”
 - Num3 for “right”
 - Ctrl + Num3 for “left”
 - (b) Select faces where you want to create a window or door. An easy way to do this is by:
 - i. Deselecting all faces by pressing A once or twice.

- ii. Pressing B to activate the Border Select tool.
 - iii. Clicking and dragging LMB to delimit a rectangular area.
 - iv. After you release LMB , all faces in the rectangular area will be selected.
- (c) Press E to activate the extrude tool.
- (d) Extrude inward 1/10th of a BU by typing $-.1$ and confirming it with Enter or LMB .

2.9.9 Final Steps

1. Adjust the position of the lamp and aim the camera until you obtain a good render.
2. Save your work!

2.10 Extruding a Simple Person



Your simple person will look like this.

In this module, you will model a simple human figure. Along the way, you will practice using extrusion and learn additional ways to select vertices, edges, and faces.

2.10.1 Start a New Scene

1. Start with the default cube (*File → Load Factory Settings*) and NumLock “on”.
2. Press Tab to edit the cube.
3. Scale the cube down 50% by pressing S . 5KEY ENTER .

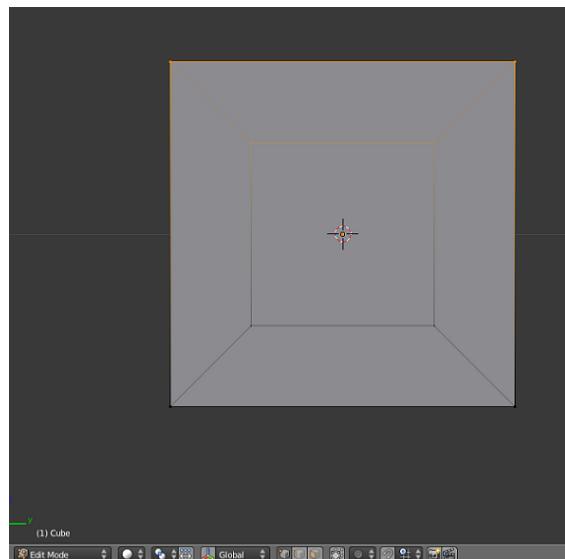
2.10.2 Selection Methods

Just as you did for the house model, you will begin by selecting the top four vertices of the cube. This section presents six methods for doing so.

Ease of selection depends partly on the viewport settings and viewpoint. For greatest ease, you want a view in which the parts you are trying to select are both visible and close together.

For clarity, use a view of the cube in which all vertices are visible:

- Go to right side view with Num3 .
- Disable the manipulator widget with Ctrl + Space .
- Make sure the Limit selection to visible option is “off”.



The picture on the right shows the cube with the correct vertices selected.

To begin, make sure you start in Vertex select mode.

Border Select Tool

The border select tool selects things that lie in a rectangular region of the viewport.

1. Activate (place the mouse pointer in) a 3D View window.
2. Deselect all vertices by pressing A .
3. Press B to activate the border select tool. Two dashed gray lines should appear, one vertical and one horizontal, forming a crosshair in the viewpoint.

4. Click and drag LMB diagonally across the area you want to select. The area will be outlined in dashed gray lines.
5. When you release the mouse button, the vertices inside the rectangle will be added to the selection.

Practice selecting the top four vertices this way. If you make a mistake, press A and try again.

Circle Select Tool

The circle select tool selects or deselects things that lie in a circular region of the viewport.

1. Activate a 3D View window.
2. Deselect all vertices by pressing A .
3. Press C to activate the circle select tool. A dashed gray circle should appear. note: Prior to Blender 2.5 B B twice.

When this tool is active, you can do various things:

- To move the select area, simply move the mouse pointer.
- To resize the select area, use SCROLL or Num+ / NUM- ..
- To select all vertices within the circle, click LMB .
- To deselect all vertices within the circle, click MMB or Shift + LMB .
- To deactivate the tool, press Esc or RMB .

Practice selecting the top four vertices this way. If you make a mistake, press A and try again.

Lasso Select Tool

Like many graphics programs, Blender 3D has a lasso select tool.

1. Activate a 3D View window.
2. Deselect all vertices by pressing A .
3. Click and hold Ctrl + LMB .
4. Drag the mouse pointer in a loop around the vertices you want to select. As you drag, a dashed gray line will appear.
5. You can deselect with lasso by pressing Ctrl + Shift + LMB .
6. Release the LMB when you're done.

Vertex by Vertex Selection

You can select (or deselect) vertices one by one, as you did in the “Quickie Model” module.

1. Click RMB on a vertex to make it the only selected vertex.
2. Toggle the select state of additional vertices by clicking Shift + RMB .

Edge Select Mode

You can select (or deselect) edges one by one, as you did in the “Improving Your House” module.

1. Click LMB on the Edge select mode button in the 3D View header.
2. Select the top left edge of the cube by clicking on it with RMB .
3. Toggle the select state of top right edge of the cube by clicking on it with Shift + RMB .
4. Switch back to Vertex select mode by clicking LMB on the Vertex select mode button in the 3D View header.

After you switch back to Vertex select mode, all four vertices in the two selected edges are selected.

Face Select Mode

You can select (or deselect) faces one by one, as you did in the “Improving Your House” module.

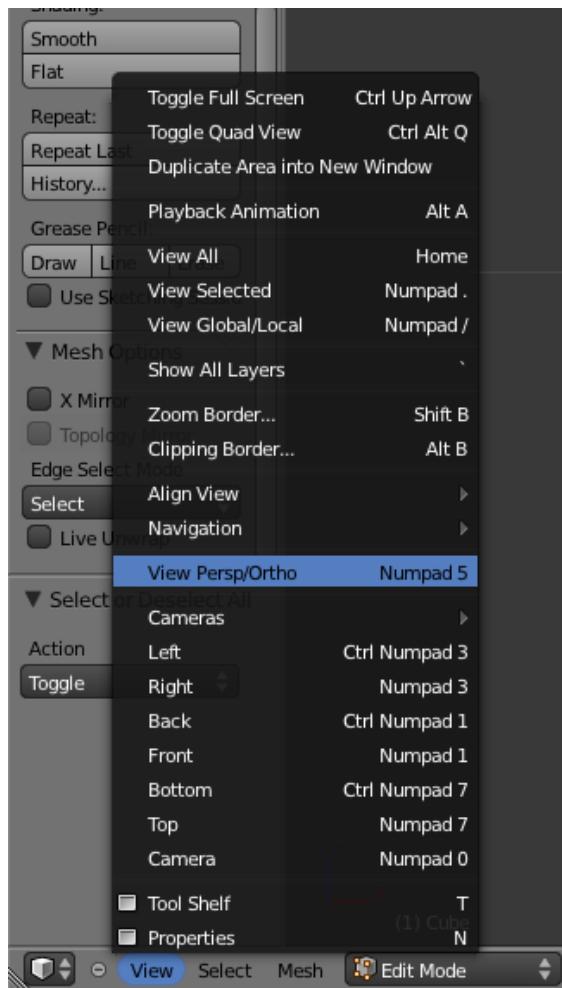
1. Click LMB on the Face select mode button in the 3D View header.
2. Select the top face of the cube by clicking on its center dot with RMB .
3. Switch back to Vertex select mode by clicking LMB on the Vertex select mode button in the 3D View header.

After you switch back to Vertex select mode, all four vertices in selected face are selected.

2.10.3 Extruding Limbs

The illustrations in this section are in front orthographic view, so:

- Use Num5 (or *View → Orthographic*) to switch to orthographic view.
- Use Num1 (or *View → Front*) to switch to front view.



The view menu.

Region Extrusion

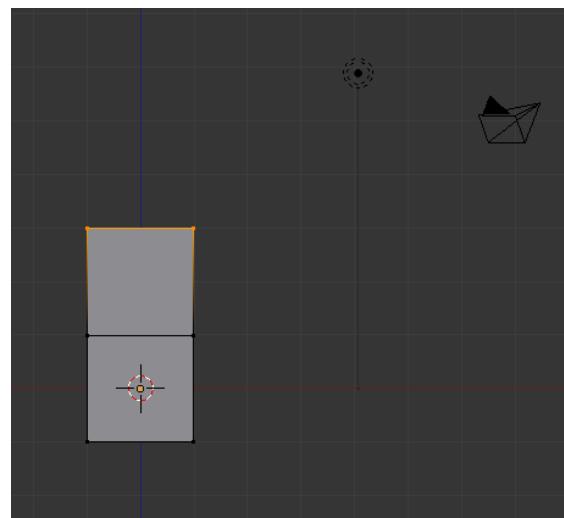
1. Make sure you're still in Edit Mode, with the top four vertices selected. (Only two will be visible in front ortho view.)
2. Activate the extrude tool by using E (or *Mesh → Extrude Region*).
3. Move the mouse pointer upwards. As you do, four new vertices will appear, each connected to one of the four that were previously selected.

The new vertices and their associated edges will move with the mouse pointer. You can lock them into place with LMB or Enter).

Extruding a Leg

Suppose you want to extrude a region the same size as the default cube -- in other words, one Blender unit on a side.

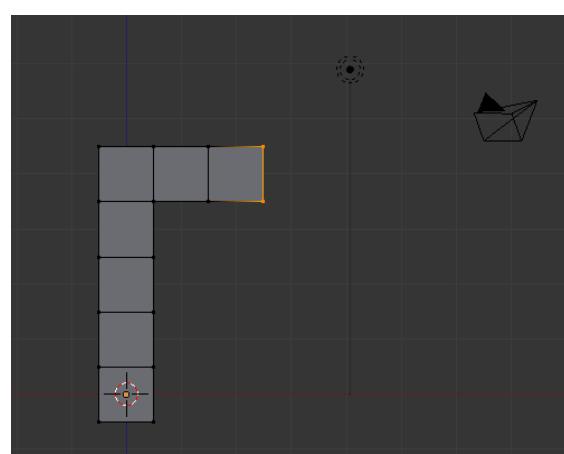
1. Undo your previous extrude by pressing Ctrl + Z .



2. Activate the extrude tool again by using E (or *Mesh → Extrude Region*).
3. This time, as you're moving the extruded vertices around, hold down the Ctrl key. You'll see that the new vertices will only move in multiples of a Blender unit. This is called snapping, and it makes it easy to extrude by exactly one blender unit. The size of the snapping depends on the zoom level; if you are zoomed out a long way from the object the snapping will be done in large increments and if you are zoomed in close you can snap in finer amounts.

Continue extruding until you have five cubes of equal size stacked atop one another. This will be one leg of your figure.

Extruding the Pelvis



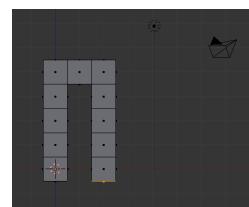
1. Press A until all vertices are deselected.
2. Rotate the view (by dragging MMB) so you can see all four vertices on the right face of the top cube.

3. Select those four vertices.
4. Extrude twice to the right.

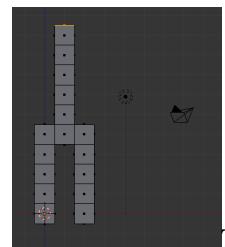
Extruding the Rest of the Body

The same trick is repeated over and over to build the rest of our simple body.

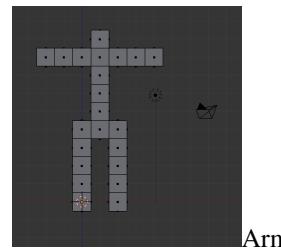
1. Create a second leg by extruding down four times from the last cube of the pelvis.
2. Create the torso by extruding up five times from the middle cube of the pelvis.
3. Extrude to each side from the next-to-top cube of the torso to create arms. (Making sure there are five on each side. Refer to the picture on the top of the page)



• Legs and Pelvis



• Torso



• Arms

To be safe, remove any double vertices you may have inadvertently created:

1. In Vertex select mode, press A until all vertices are selected.
2. With a 3D View window active, press W to bring up the *Specials* menu.
3. Choose *Remove Doubles*.

Now check your work:

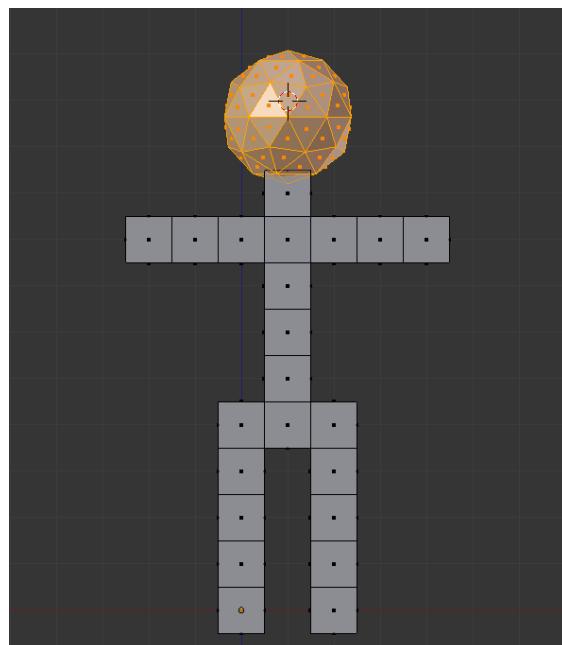
1. Return to Object Mode by pressing Tab .
2. Make sure the viewport draw type is Solid. (Press Z if it isn't.)
3. Rotate the viewpoint and examine the body from every side (it might be useful to return to perspective view for this).

If any faces are missing...

This is easily fixed. To create a face:

1. Press Tab to go back into Edit Mode.
 2. Select four vertices.
 3. Press F (or choose *Mesh → Faces → Make Edge/Face* from the 3D View header).
- Note that you can also make edges with this tool if you select two vertices.

2.10.4 Adding the Head



1. Move the 3D cursor to a point above the neck by clicking with the LMB .
2. Adjust the cursor position in orthographic top, front and side views (Num7 , Num1 , and Num3 respectively) until the 3D cursor is about where the center of the head should be. It may help to use Shift + S → *Snap → Cursor to Grid*.

3. Make sure you're in Edit Mode with a 3D View window active. (If you create the head in Object Mode, it will be a separate object from the body, and changes to the body later in this tutorial won't affect the head.)
4. Create a sphere using Shift + A → Mesh → Ico-sphere.
5. Leave the default settings for subdivisions and size in the bottom left of the screen. (Note: Your computer may slow down if you set subdivisions above 6)

You should now have a small sphere at the top of the body. To make it more proportional to the body, resize it using the scale tool:

1. Make sure you're still in Edit Mode, with a 3D View window active and the head selected.
2. If necessary change the pivot point to *Median Point*.
3. Activate the scale tool by pressing S (or *Mesh* → *Transform* → *Scale*).
4. Move the mouse pointer until the head is the size you want.

You may also adjust its position using the grab tool:

1. Make sure you're still in Edit Mode, with a 3D View window active and the head selected.
2. Activate the grab tool by pressing G (or *Mesh* → *Transform* → *Grab/Move*).
3. Move the mouse pointer until the center of the head is where you want it.

Now check your work:

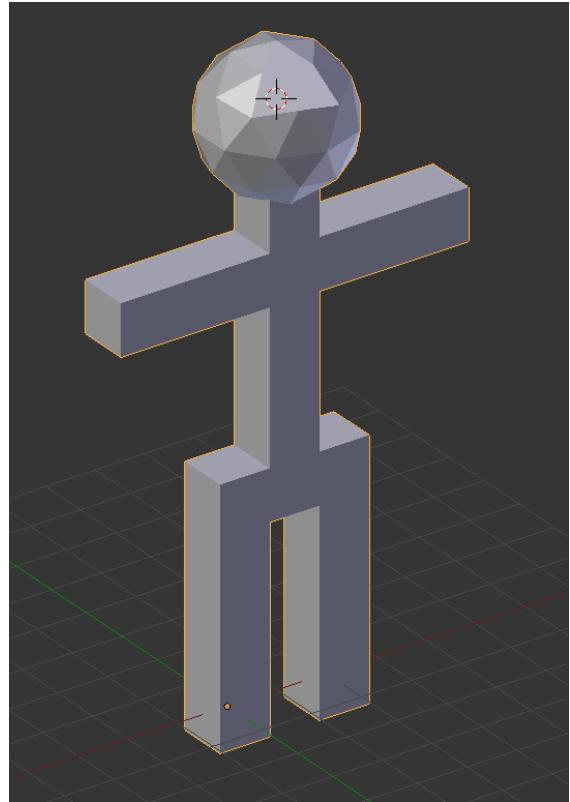
1. Return to Object Mode by pressing Tab .
2. Make sure the viewport draw type is Solid. (Press Z if it isn't.)
3. Rotate the viewpoint and examine the body from every side. Make sure that the head connects properly to the neck.

2.10.5 Save Your Work

You will continue working on your simple person model in the next module.

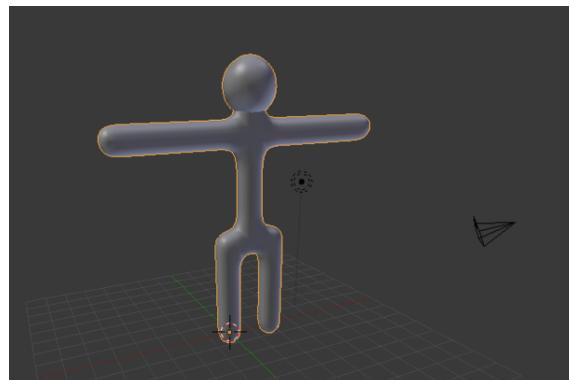
To save the scene in a .blend file:

1. Press ctrl + S (or select *File* → *Save*).
2. Navigate to the directory (folder) where you want to write the file.



3. Type a filename in the text box to the left of the “Cancel” button.
4. Click LMB on the “Save Blender File” button.

2.11 Smoothing Your Simple Person



Your goal.

Few real-life objects have perfectly sharp edges. People, in particular, consist of mainly smooth surfaces. How does one model a smooth object using flat faces and sharp edges?

In this module, you'll learn how to smooth a mesh by using subsurfaces and smooth shading.

You'll need the simple person model from the previous module. If you haven't done it, either go back and do it now or download the pre-made model from Yoson Chang's website at <http://www.nusoy.com/blender>.

If the model doesn't look solid, your Viewport Shading setting may be set to Wireframe. To switch to Solid shading:

1. Activate the 3D View window.
2. Press Z .

2.11.1 Subsurfaces



The sub surface modifier.

So far, all the meshes you've created have had sharp edges, giving them a faceted appearance like that of a cut diamond. To model a smooth object (like a human body) you might think you need a huge number of vertices and faces. Subsurfaces partly solves this problem by automatically subdividing a mesh into a finer mesh suitable for smooth rendering.

You subsurface in Blender by adding a subsurf modifier to an existing mesh object. A modifier is simply an algorithm (automatic process) which can be added to an object. (Blender modifiers are analogous to Photoshop adjustment layers.)

To get started, make sure Blender is in Object Mode, with only the simple person object selected:

1. If Blender is in Edit Mode, press Tab .
2. To select the simple person, RMB on it.

To add a subsurf modifier to the selected object:

1. Click on the modifiers tab (wrench icon) in the Properties window.

2. Add Modifier → Subdivision Surface.

You could also add subsurf modifier by pressing Ctrl + 1Key .

The object's appearance should immediately become more faceted and more rounded. In addition, several sub-surface controls will appear in the Modifiers tab.

If a few faces don't subsurf...

The model may include some double vertices. To get rid of these:

1. Edit the model in Vertex select mode.
2. Select all vertices.
3. Mesh → Vertices → Remove Double
4. Try again.

If Blender crashes when you attempt to subsurf an object...

You need to look in to upgrading (or possibly even downgrading) your graphics drivers. Having the right graphics driver can avert many problems.

What just happened? The default subsurf modifier (one level of Catmull-Clark) subdivided each face of the object into four smaller faces that are progressively angled. This softened the sharp edges of the original model where faces met at 90-degree angles.

Controls

For this model, one level of subsurf isn't quite enough. To increase the number of levels to two, just increase the number in the text box directly underneath *Subdivisions*. The *View* setting controls the number of subdivision levels visible in the viewport. This is very useful when you have a high-poly scene, just decrease the number of visible subdivisions to speed up viewport action.

You can specify additional levels of subsurfing to be used during renders. For extra smooth renders, you might want three levels of subsurfing. Set this with the *Render* control immediately below the *View* control.

The *Apply* button applies the modifier to the mesh. Do not click it yet. We'll be playing with the model a bit longer before we apply the changes. While useful with some modifiers, applying a subsurf modifier produces a very complex mesh, and there's no need to do so here.

Remember that you can undo any accidental modifications with Ctrl + Z .

Blender can combine a series of modifiers by stacking them. For this reason, the Modifier tab includes buttons for arranging and removing modifiers.

You can hide edges created by the modifier by activating the *Optimal Display* toggle button. The effect is especially clear with the Wireframe draw type.

You can edit the mesh in modified form (without actually applying the modifier) by activating the *Adjust edit cage to modifier* toggle button, a small button with a triangle and vertices, to the left of the Up/Down arrows (the arrows are for changing the position of the modifier in the stack)

in the Modifier panel.



Try this out:

1. Press Tab to enter Edit mode.
2. Make sure Blender is in Vertex select mode.
3. Activate the *Adjust edit cage to modifier* button.

Now all vertices lie on the surface of the object, and you can adjust the (modified) vertices directly. However, any additional vertices created by the modifier cannot be directly edited without applying the modifier.

You will be editing the boxy version of the simple person awhile longer, so before continuing, deactivate the *Apply modifier to editing cage during Editmode* button.

2.11.2 Smooth Shading



Your simple person after setting smooth.

Subsurfaces do a good job of smoothing out corners in meshes. Even with two levels of subsurfaces, however, the simple person does not look completely smooth; when viewed close up, it has a scaly appearance. This is because each face is flat shaded—shaded to resemble a

flat surface—resulting in sudden changes in brightness at most edges. For a smooth object, you want smooth shading, which smooths out the changes in brightness.

1. Go to Object Mode.
2. Set the draw type of a 3D View window to “Solid”.
3. Select your subsurfed object.
4. In the Toolshelf on the left, look for a caption called *Shading*. Under it should be a button called *Smooth*.

All the mesh edges will be smoothed out, leaving no sudden changes in brightness. The faces blend smoothly into one another, making the edges nearly invisible. If the icosphere has not smoothed properly and is dimpled, enter Edit Mode by pressing Tab, select all vertices (A) and recalculate the normal direction (CTRL+N). This is also available in the Toolshelf under *Normals*.

5. Click the other button under *Shading* in the Toolshelf, named ‘Flat’.
- The edges will reappear. Now you know the difference between Flat and Smooth.
6. Since the model looks better with smooth shading, click LMB on the “Smooth” button again.

Note that if you didn't have subsurf enabled, then the mesh wouldn't look much different. This is because smooth shading doesn't affect the mesh shape, it just changes how the computer draws the triangles.

Smooth shading also removes a lot of definition. A good way to get rid of this is simply to add a subsurf modifier like you just did. The modifier will not only require fewer vertices, but add definition.



• Flat Shading



• Smooth Shading

Save your work. You will continue refining this model in the next module.

Additional Resources

-  Flat shading at Wikipedia.
-  Gouraud shading at Wikipedia.
-  Phong shading at Wikipedia.
-  Subdivision surface at Wikipedia.
- For more about modifiers, see the Blender Manual page on “The Stack” at http://wiki.blender.org/index.php/Doc:Manual/Modifiers/The_Stack
- For more about Blender subsurfaces, see the Blender Manual page on “Subdivision Surfaces” at http://wiki.blender.org/index.php/Doc:Manual/Modeling/Meshes/Subdivision_Surfaces.

2.12 Improving Your Simple Person

In this module, you'll edit a subsurfed mesh using the scale and grab tools, all the while improving your character.

You'll need the simple person model from the previous module. If you haven't done it, either go back and do it now or else download the pre-made model from Yosun Chang's website at <http://www.nusoy.com/blender>.

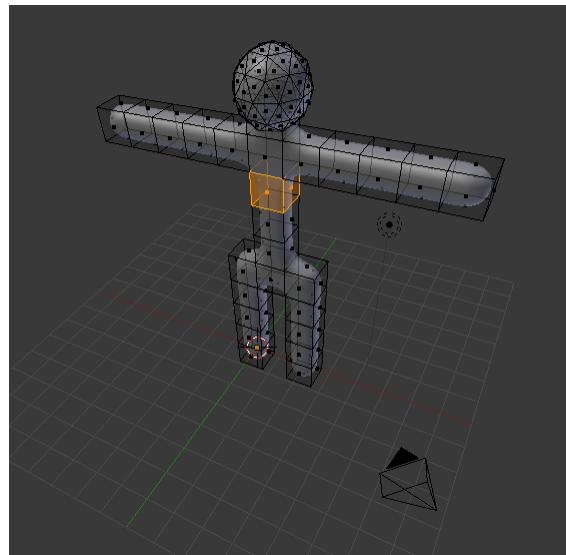
2.12.1 Widening the Torso

To be realistic, the simple person's torso needs to be three times wider. In order to keep the torso symmetrical, you'll expand it by scaling both sides from a central point.

Select the sides of the torso:

1. Enter edit mode on the simple person.
2. In the 3D View header, set Face select mode.
3. From the 3D View header, choose *Pivot → Median Point*.
4. In the 3D View header, make sure Proportional Edit button is off.
5. Select the two faces on both the left and right sides of the torso, between the armpits and the waist.

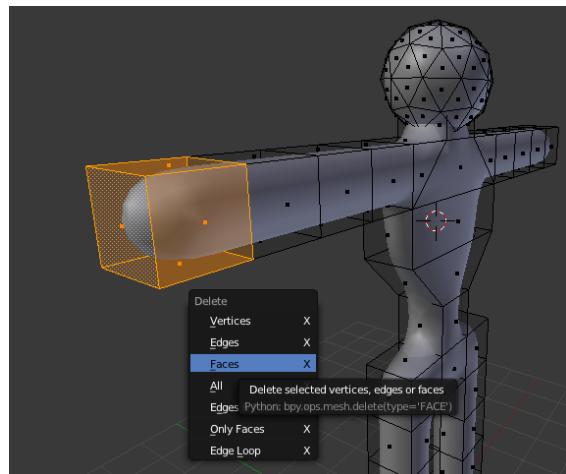
We will now scale the torso with the scaling tool:



1. Activate the 3D View window and press S , X .
2. Adjust the amount of scaling. Either:
 - Move the mouse pointer until the torso is the width you want.
 - or
 - Press 3
3. Confirm and exit by pressing Enter or clicking LMB

Continue selecting different parts of the torso and scaling them to get more practice using the above scaling methods.

2.12.2 Bending the Arms



Removing the forearm

When you've got the basic shape of the torso, make the person hold up his hands. You'll do this by deleting the forearms and then extruding upward from the elbows.

Select both forearms:

1. Enter edit mode on the simple person.
2. In the 3D View header, set Face select mode.
3. With the 3D View window active, press A until all vertices are deselected.
4. Select the five faces at the end of the forearm.

Now erase them:

1. Press X to open the Delete menu.
2. Choose *Faces*.

The forearm will disappear, leaving a hole. Don't panic; we'll fix it later. Now to make the arm point upwards:

1. Select the top face of the last remaining "arm cube".
2. Extrude the region upward by two Blender units E , Z , 2 and confirm with LMB or Enter .

(Newbie comment: on my system, using Blender 2.70a, **you want E-2 above, not E-Z-2**. Z-axis constraint is on by default, so pressing Z turns it off and causes trouble. Confirmed by second newbie in Blender 2.78.4.)

The hole in the elbow is caused by a missing face. To fill in the missing face:

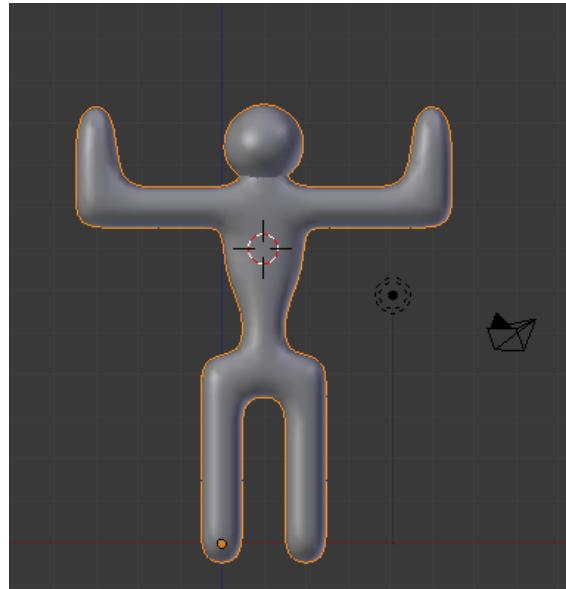
1. Deselect all vertices.
2. Select the four vertices surrounding the missing face.
3. With the 3D View window active, create the face using either
 - *Mesh → Faces → Make Edge/Face*

or

- F

The new face should be smooth. If it isn't, make it so, using *Mesh → Faces → Shade Smooth*.

Go through the same steps (erase, extrude, and fill) on the other arm. Be sure to deselect all vertices in the first arm before selecting any in the other arm. If you have difficulty making the arms symmetrical, undo your work and go through the steps simultaneously on both arms.



Repeat on the other side

2.12.3 Making Feet

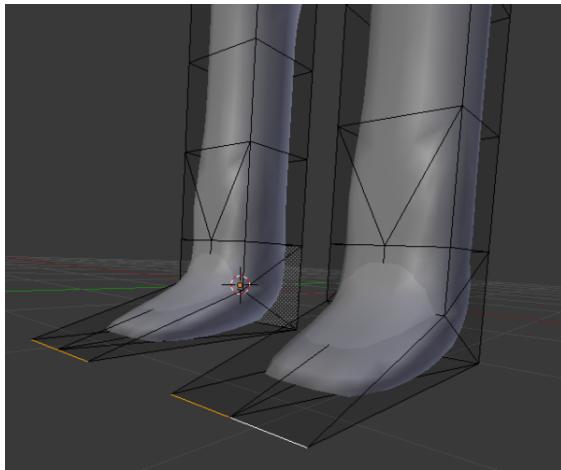
To make feet for your simple person, you subdivide the ends of the legs and pull the front edges forward.

1. Edit the simple person in Face select mode.
 2. Select the two bottom faces of the legs (front of the feet) by clicking RMB on the first and then Shift + RMB on the other.
 3. Subdivide both faces, either with:
 - *W Subdivide*
- or
- *Mesh → Edges → Subdivide*

Each face gets subdivided into four smaller faces.

Now select the front edges and pull them forward:

1. Switch to Edge select mode.
2. Press A until no edges are selected.
3. Select the four bottom front edges of the soles (two for each foot) (where the toes should be).
4. Press G and limit movement to the Y axis.
5. Move the mouse pointer until the feet are the length you want.
6. Confirm and exit by pressing Enter or Space or clicking LMB .



Congratulations! You now have feet.



2.12.4 Reshaping the Head

When you're satisfied with the torso and limbs, you should do something about that head. A bit too spherical, isn't it? You can elongate it by scaling along the Z axis.

When scaling the head, you want to make sure that it stays connected to the neck.

First, place the 3D Cursor at the base of the head, where it meets the neck. An easy way to do this is as follows:

1. Go into Vertex select mode.
2. Make sure the *Limit selection to visible* option is “off”.
3. Select the vertex at the base of the head using RMB .
4. Snap the cursor to this vertex using Shift + S *Cursor to Selected*

Now select the entire head:

1. Hover the mouse over a vertex/edge/face of the head
2. Press L to select all parts linked to that part.

Tell Blender that you want to pivot around the 3D Cursor by changing the pivot point to *3D Cursor* on the Pivot menu (the small button located to the left of the 3D Manipulator button).

Now scale the head along the Z-axis, using the scale tool (S , scaling by 1.5 should be about right).

You'll need this simple person later, so remember to save your work!

2.13 Spinning a Simple Hat

In this module, you'll create a hat for your simple person. Along the way, you'll learn how to use the Spin tool and use layers.

2.13.1 Creating a Generatrix

For future convenience, you'll create the hat as a new object in the scene containing the simple person. If you haven't created the simple person, either go back and do it now or else download the pre-made model from Yosun Chang's website at <http://www.nusoy.com/blender>.

Start by changing layers to layer two, then add the basis for your hat:

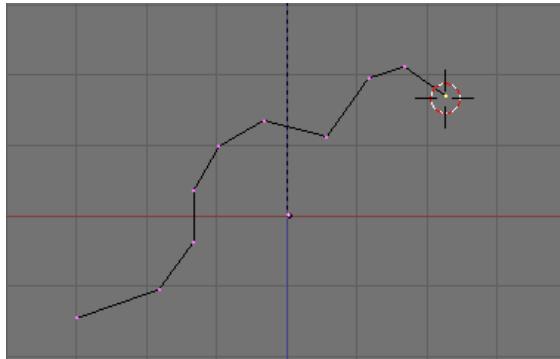


1. Make sure you're in Object Mode (so that a new object will be created).
2. Click LMB on the second little square, this will make the viewport display layer two. (The top row is for layers 1 to 10, the bottom for 11 to 20, so layer 2 is immediately to the right of layer 1; layer 6 is across the space from layer 5.)
3. Go to orthographic front view by pressing Num1 .
4. Create a mesh circle at the cursor, by activating the 3D View window, pressing shift + A and choosing *Mesh → Circle*.

The new mesh object doesn't actually have to be a circle. You could use any sort of mesh object here because you're about to reshape it into a custom 2D mesh (called

a generatrix) that describes the profile of your hat. More precisely, the generatrix describes one side of a vertical cross-section through the hat. You'll want your generatrix to have a slope; it should be higher on one side (which will become the top of the crown) than on the other (which will become the brim).

1. The newly-created mesh should be selected. If it isn't, select it by clicking RMB on it.
2. Press Tab to edit the mesh.
3. Activate Vertex select mode.
4. Press A until all vertices are selected.
5. Press X to erase all vertices.



Now draw your generatrix, starting with the brim and sloping upwards toward the top of the crown:

1. Make sure you're still in orthographic front view.
2. Press Ctrl + LMB to create the first vertex.
3. Press Ctrl + LMB to one side of that vertex to extrude another vertex, connected to the first by an edge.

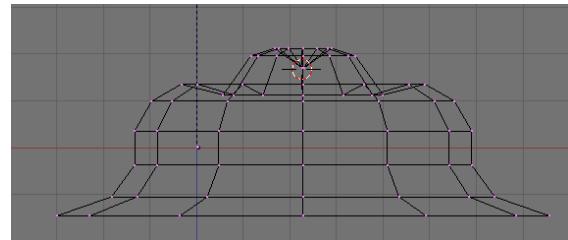
(If this doesn't work, make sure you are in vertex select mode.)

Keep adding vertices until you're satisfied with the shape of your generatrix. You can always undo using Ctrl + Z or go back and adjust the positions of particular vertices using the grab tool.

The mesh is then spun around an axis perpendicular to the viewplane. You want to spin around a vertical axis, so press Num7 to switch to top view.

2.13.2 Spinning the Hat

Now, let's actually spin the hat:



The spun hat, drawn as wireframe in orthographic front view.

1. Move the 3D cursor to the vertex you want to spin around by pressing LMB on it. You can also use the snapping tool for positioning the cursor more precisely by pressing Shift+S after selecting that specific vertex. *Cursor to selected* positions the cursor.
2. Press A to select all the vertices. The Spin control only spins vertices that are selected.
3. Press Alt+R to activate the Spin tool.
 - The Spin tool is also available in the Tool Shelf under *Add*

If you spin the hat in front view, your hat will be flat. You have to spin the hat in top view.

You should now see 90° of a generatrix! To spin your hat all the way round, press F6 or look in the Operator Panel just below the Tool Shelf. There should be an input slider named *Angle*, change this value from 90 to 360. There should also be a slider called *Steps*, increase the value from 9 to 15.

If your hat has a large hole in the center, you must have accidentally moved the 3D cursor away from the vertex you picked in step 1. Try again.

Remember that if you spin an object 360° there will be a double row of vertices at the row of vertices you spun. To fix this, press A to select all vertices, press W and select *Remove Doubles*. Note that this will only work in vertex select mode.

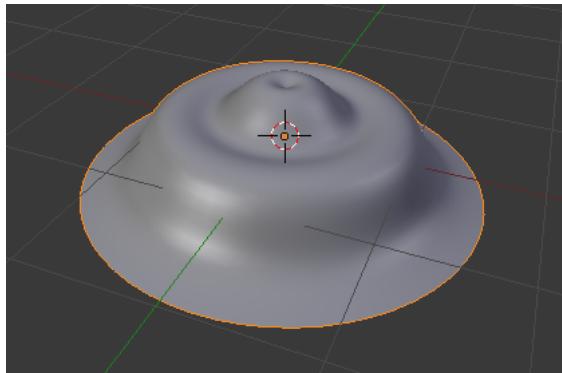
You may also want to merge the vertices at the top of the hat. Do this by selecting all the vertices at the top with C and pressing Alt + M → *At Center*. You may have to do this twice as some vertices might be beneath each other.

If the mouse pointer changes to a question-mark (?)...

You have more than one 3D View window, so Blender is asking which window to perform the spin in. Click LMB on the window that is showing top view.

2.13.3 Smoothing Your Hat

You'll probably have noticed that normal hats aren't usually as faceted as yours! To change this, first press Tab



The finished product!

to go back to Object mode then change the shading to *Smooth* (available on the Tool Shelf). If there are unexpected black marks, try recalculating the normals.

1. Switch to Edit mode and open the Mesh menu in the 3D View Header.
2. *Normals → Recalculate Outside.*

Next, add a Subsurf modifier to the hat and set the subdivisions to two, as you did in the “Detailing Your Simple Person 1” module.

1. Click on the modifiers tab (wrench icon) in a Properties window.
2. *Add Modifier → Subdivision Surface.*

Save your work. You'll need this scene for the next module.

2.13.4 Additional Resources

-  Surface of revolution at Wikipedia.
- http://www.youtube.com/watch?v=-mPtxa_MEPA Ira Krakow's Hat Creation Video Tutorial, based on this page.

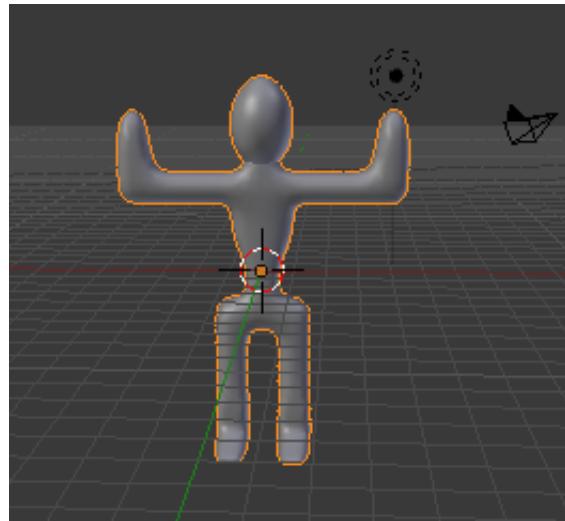
2.14 Putting the Hat on the Person

Once you're satisfied with the shapes of individual objects, you'll want to combine them into a coherent scene. You do this in Object Mode.

In this module, you'll learn how to move objects to and from layers. You'll also learn how to rename and parent objects, and you'll get an introduction to Outliner Windows.

You'll need the person-and-hat scene from the previous module. If you haven't done it, either go back and do it now or else download the pre-made model from Yosun Chang's website at <http://www.nusoy.com/blender>.

2.14.1 Adjusting an Object's Median Point



The person that you (yeah you!) made with the origin in his geometric center.

1. Load the person-and-hat scene.
2. Make sure Blender is in Object Mode.
3. Switch to Layer 2, select the hat and press M . A dialog box will pop up for you to choose which layer to move it to. Either press 1 (the number on top of the keyboard, **not** the numberpad) or select the first box in the popup.
4. Select the person you made earlier.

Just as you did in Edit Mode, you can specify the pivot for rotating and scaling objects in Object Mode. If you just finished the previous module, the pivot is probably set to “3D Cursor”. If so, change it back to “Median Point”.

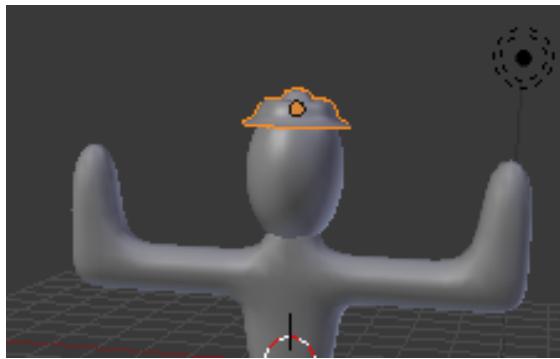
In Edit Mode, the “Median Point” for pivoting is the geometric center of all selected vertices, edges, or faces. In Object Mode, however, it's the origin of the selected object's local coordinates, indicated by an orange dot. In other words, the origin might lie far from the object's geometric center.

You can use buttons in the Tools Shelf to reunify an object's origin with its geometric center:

1. With Blender in Object Mode, click LMB on *Set Origin* in the Tool Shelf (under the “Edit” sub menu of *Tools*) and select *Origin To Geometry* (Blender 2.70: “Object” -> “Transform” -> “Origin to Geometry”) to move the selected object's origin to its geometric center (without changing the object's appearance).

This can be useful when you want a better picture of your object. With the origin set to the person's geometric center, you can now snap the object with Shift+S to the 3D cursor. This will let you view more of the model at one time and make for a faster editing workflow.

2.14.2 Positioning the Hat



Positioning the hat

Once you have the hat properly oriented, move it into position on the person's head. The grab tool enables you to position objects in Object Mode in the same way you positioned vertices, edges, and faces in Edit Mode.

1. Make sure Blender is in Object Mode.
2. Click RMB on the hat object to select it.
3. Activate the grab tool by pressing G .

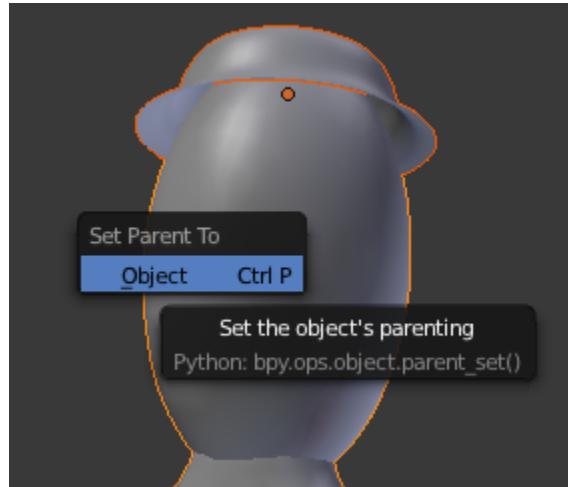
As you move the mouse pointer, the hat will move around in the viewport. By default, the movement plane is perpendicular to the view axis, so the hat will move differently depending on which viewpoint you're working in.

Just as in the Edit Mode grab tool, you can:

- Restrict the direction of motion by pressing X , Y , or Z . Press once to move parallel to a global axis, twice to use a local axis. (Press the same key a third time to return to view-plane motion.)
- To restrict motion to the global X-Y plane, lock the global Z by pressing Shift + Z .
- Hold down Ctrl to restrict motion to discrete steps (typically one Blender unit).
- Hold down Shift to get finer control over the motion.
- Click LMB or press Enter to finalize the position and exit the tool.
- Click RMB or press Esc to return the object to its previous position and exit.

Use two different orthographic views to position the hat on the person's head. You will probably want to scale the hat to make it fit the person's head better. When you are doing this along the X or Y axis, make the changes symmetrical by specifying the axes you want scaling to be constrained to. This option is available in the Operator panel (just below the Tool Shelf) and also by pressing F6.

2.14.3 Parenting the Hat to the Person



The parenting menu.

Once you have the hat properly sized and positioned on the person's head, you'll want it to stay there. In order to maintain such a cozy relationship between two objects, you'd have to remember to select them both before rotating, moving, or scaling. A drastic solution might be to join them into a single object using Ctrl + J .

A better compromise is to Parent the hat to the person. Parenting creates a relationship between two objects, such that certain changes to one object (called the Parent object) automatically affect the other (called the Child object). Changes to the child, however, do not affect the parent.

Note that an object can have many children, but only one parent.

Since the person is bigger than the hat, it's logical to parent the hat to the person (meaning: parent = person, child = hat) instead of vice versa.

1. Make sure Blender is in Object Mode.
2. Click RMB on the hat object to select it.
3. Click Shift + RMB on the person object.
4. Press Ctrl + P to parent the hat to the person.

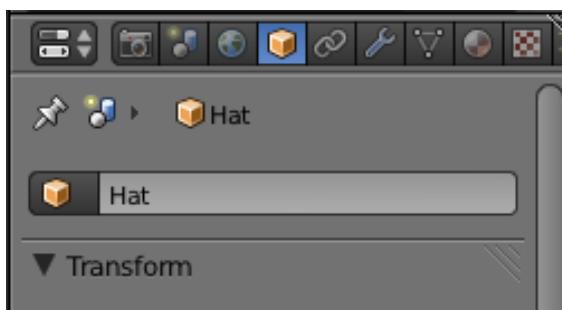
Both the person and the hat should now be selected. The order of selection is important here.

5. Select *Object*. The most recently selected object becomes the parent of all other selected objects.

Now when you move the hat you will see a line from the hat to the person, indicating that the person is the hat's parent. And if you move the person, the hat will move with it.

You may get an error saying something like *Loop to Parents*, fix this by clearing all previous parents with Alt + P

2.14.4 Renaming Objects



The renaming dialog

When you have multiple objects in a scene, it helps to give each one a name.

Click on the *Objects* tab in the Properties panel (the one with a box icon).

1. Now select the hat by clicking RMB on it.
2. At the very top of the tab you should see a dialog box with the name of your object
 - The hat's name might be something like "Circle" depending on which mesh primitive you first built the hat from.
3. Click LMB on the dialog box and type in a more descriptive name like "Hat".

You have now changed the name of the hat's object datablock. This name change will be reflected in the Outliner, which we will look at shortly.

Now select your person by clicking RMB on it and repeat the process, changing the name to something like "Person".

2.14.5 Outliner Windows

Once you give objects names, it helps to have a way to find objects by their name and parent. This is exactly what the Outliner is for and it comes in very handy when you are working with a large scene. The Outliner is usually just



The Outliner window.

above the Properties panel. You may want to pull it down a bit to see it more clearly.

You'll notice that all the objects in your scene (Person, camera etc) are listed and that you can select these objects by clicking LMB on them. And if you click RMB on an object, a menu will pop up with options like *Select*, *Deselect*, *Delete* etc. If you select the Person and then click the "+" sign to its left, you will see that the Hat is listed below the person. This is because Blender lists all children objects beneath their parents.

On the right of each object there are a series of icons which represent the state of the object. For example, the eye icon means that your object is visible in the 3D viewport. You can turn off its visibility by clicking LMB on the eye, which will turn grey; click again on the eye to make it visible. If you hover the mouse over the icons a text box will pop up with a description of what that particular icon does.

2.14.6 Good on ya' mate!

Congratulations!! You have now finished your simple character. Pat yourself on the back, and have a celebratory coffee! (Or pop!)

2.14.7 Additional Resources

- Outliner at <http://www.blender.org/development/release-logs/blender-235a/outliner/>
- Gestures at http://www.blender.org/manual/modeling/meshes/editing/basics/translation_rotation_scale.html

- Parents at <http://www.blender.org/manual/editors/3dview/object/relationships/parents.html>

2.15 Overview

In 3D graphics, materials and textures are nearly as important as shapes. Scenes would be boring if all the objects were gray.

The material system in Blender allows you to model a wide variety of materials and how they interact with light. The next few modules will introduce the available options.

2.15.1 Material versus Texture

A material defines the optical properties of an object: its color and whether it is dull or shiny. A texture is a pattern that breaks up the uniform appearance of the material. Very few objects in the real world have completely uniform surfaces. Instead most of them have patterning or variation in color: consider the grain in a piece of wood, the pile in a carpet, or the mortar in a brick wall.

Blender allows textures to influence materials in various ways, such as altering their colors. Multiple textures can interact with each other to produce interesting effects.

Note that textures have to be attached to materials to affect objects, you cannot apply a texture to an object without a material.

2.15.2 Other Material Settings

Additional settings you can specify for a material include shaders, ray-tracing and halo.

Shaders determine how the appearance of a material varies with the angle of the light: diffuse shaders give a non-shiny look, while specular shaders give a mirror-like finish. Blender's material settings always involve both kinds of shaders, but you can adjust a material's diffuse and specular colours separately to control their respective effects; if you set the specular colour to black, the surface will no longer produce reflections.

Ray-tracing is a technique for modeling the physical path of light through the scene. It is capable of producing exquisite reflection and refraction effects, including different degrees of reflectivity, translucency and transparency, and representing materials with different indexes of refraction. Blender provides two separate groups of ray-tracing settings, one for reflection of light and the other for its transmission through the material. You can control these settings on a per-material basis.

Halo rendering means an object no longer looks like solid matter, instead it appears to be made of bits of light. This can be used for real-world effects like fire, smoke and

plasma, or to create fantasy effects with no connection to reality.

Note that reflections produced by ray-tracing are separate from that produced by the specular shader: the former are controlled by the material's mirror colour, while the latter is controlled by its specular colour.

Reflection is done in two different ways because, while ray-tracing produces the most realistic renders, it is also very CPU-intensive. It is therefore best to apply the ray-tracing effects when you're completely done with your modelling to help reduce high CPU usage. Enough practice with ray-tracing can also help you get stunning effects with just few clicks without you having to do much trial and error. You would do well to dedicate at least a few hours of your time to experimenting with it, so that in a future real production situation, you will be spared all that hassle.

2.15.3 Types of Textures

When you create a texture in Blender, you will see a popup menu listing a whole lot of different types for the texture. The Image or Movie texture type lets you use a scanned image to texture your object: for example, you can scan an actual piece of metal and use that to give your object a realistic metallic appearance, or use a photograph of an actual brick wall to texture the wall of a building model, and so on. You could even use a movie, which plays during the animation of the scene.

The other texture types are called procedural, which means the textures are generated according to algorithms built into Blender itself. These can be useful for simulating various effects when you don't have an image of the real material handy; they can also be applied to augment the appearance in various ways. For example:

- using a “cloud” texture to “dirty-up” a material
- or
- using one texture as a stencil to create an amalgam of two other textures.

2.15.4 Additional Resources

-  Diffuse reflection at Wikipedia.
-  Ray tracing (graphics) at Wikipedia.
-  Specular reflection at Wikipedia.
-  Texture mapping at Wikipedia.

2.16 Quickie Material

In this module, you will create a new material called “Green Ooze”. Along the way, you will learn how to alter the diffuse, specular, and mirror colors of a material.

2.16.1 Your First Material

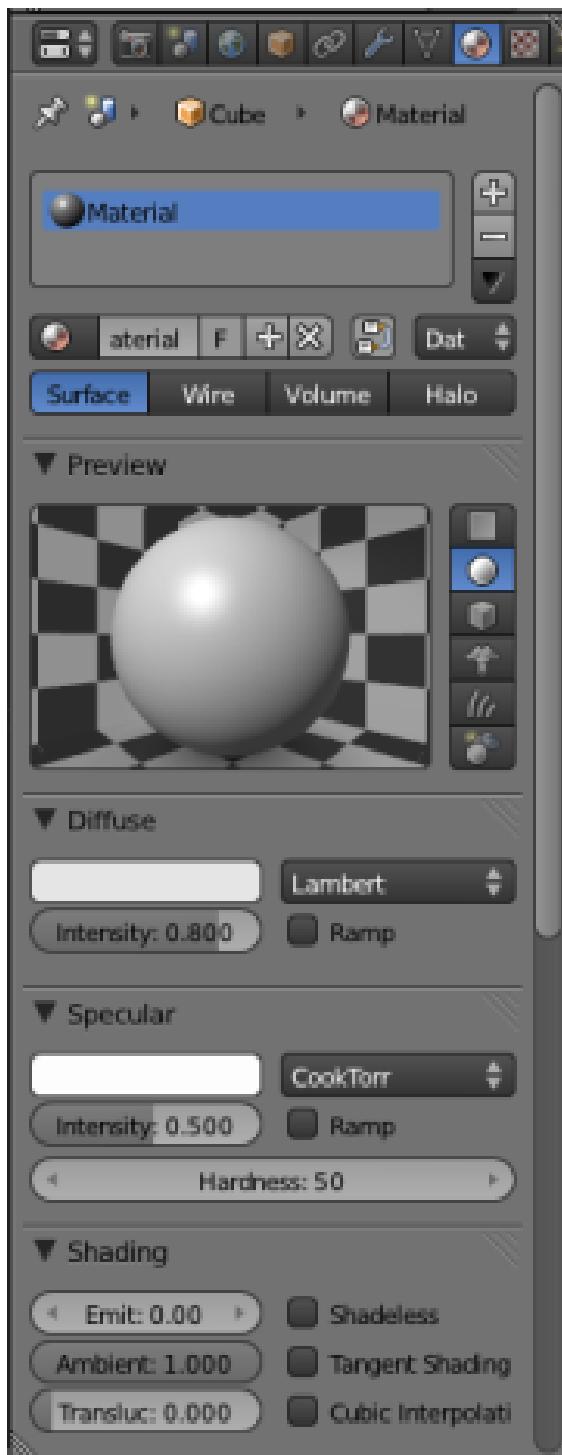


Figure 1: The Materials context in the Properties window.

The cube in the default scene (which you get from *File → Load Factory Settings*) has a simple grey color. Now click on the Materials context in the Properties window.

The materials context contains various menus, but for now you only need diffuse, specular and mirror. The material is named and linked in the panel above the preview window. (Linking is a feature that allows materials to be shared between multiple objects (or datablocks). Changing a material affects the appearance of everything it is linked to.)

The first row of the window above the preview window indicates that:

- there is one material assigned to this object and its name is “Material”.

The second row of controls indicates that:

- The current selected material’s name is “Material”.
- This material will only be saved if it’s in use.
- It is not a “Nodes” material.
- Instead of being linked directly to an object, the current material is linked to a datablock.

To rename the material, click LMB on the name and enter the name you want.

To unlink the material, click LMB on the **X** button to the right of the material name (“Material”). Do this now. This deletes the link to the datablock, removing the material from the mesh. As a side-effect, most of the panels in the Material context disappear. You will see in a moment, however, that the material still exists. It hasn’t been deleted; it is simply no longer in use.

At this point, you could click LMB the “New” button to create a new material, but instead we are going to reapply the old material:

1. Click LMB on the button to the left of the “New” button.
2. You’ll see a nifty drop-down list containing all materials you’ve created so far. Choose *O Material*.

Materials whose names are preceded by “O” in this list are not in use. By default, Blender doesn’t save such materials when it saves the scene. Thus, you can delete a material from the list by saving the scene and then reopening it. You can override this behavior by toggling the “F” button “on” for unused materials you want saved.

Your materials will be much easier to find and manage if you give them brief, descriptive names you can recognize

at a glance. Change this one's name to "Green Ooze". In addition, naming of your materials and other objects in your scene is useful when such components of your scene will be appended in another scene of a different Blender file. Naming your materials and other stuff in the scene will enable you to choose the right objects and materials you need whenever you wish to append just a portion of a whole bunch of work you did. For instance, you're working on a new Blender project, but felt the material you used in this Blender file is worth it. Instead of going through the pain of creating a new material (of course you guessed in the initial one in getting the right material appearance), you just append the material to your new work. Pretty simple! Make naming a habit, as it's much used in a production environment.

2.16.2 Specifying Colors

Simple materials are specified by three colors: diffuse, specular and mirror. Rectangular patches (*swatches*) of the colour in their own panel in the Material context allow you to see and change each of these. Diffuse color is the basic underlying color of the material, rendered by the diffuse shader. Specular color is for highlights (small bright spots on a shiny surface) as rendered by the specular shader. Mirror color is for true reflections rendered using ray-tracing.

There are many ways to define colours. Blender supports three:

- **RGB:** By specifying relative amounts of red, green and blue *primary colours*, by giving a number from 0.0 to 1.0 for each component. For example, $(R, G, B) = (0, 0, 0)$ specifies black (no colour at all); $(0, 1, 0)$ is full green; $(1, 1, 0)$ (full red + full green) is yellow; $(0.5, 0.5, 0.5)$ is 50% grey, and $(1, 1, 1)$ is full white (maximum intensity of all components). Note that this is *additive mixing* of colours, which is what happens when you shine lights of different colours onto a white screen, not the *subtractive mixing* that takes place when you mix different-coloured paints or inks on paper or canvas.
- **HSV:** By specifying a *hue* (colour position on the rainbow) together with a *saturation* (strength of colour, from garish down to pastel, with zero giving shades of grey) and *value* (brightness). This is generally considered to be easier to use than RGB notation when you are trying to create new colours (as opposed to copying a colour spec from somewhere else), since it is easier to predict what the likely result will be. HSV is commonly represented on a *colour wheel*, where the hue is the angle around the circle, saturation the distance from the centre, and value controlled by a separate brightness slider (as shown below).

- By specifying a 6-digit hexadecimal number. This is just an alternative form of RGB notation, commonly used for colour specifications in Web pages.



Figure 2: Blender's colour picker popup

If you click on any colour swatch, the colour picker will pop up, allowing you to change the values. This is the most intuitive way. The window that appears will look like this and will include the following (Figure 2):

1. A color wheel to change the color as you want. In HSV mode, H corresponds to angle around this wheel, while S corresponds to distance from the centre.
2. Three color sliders that will change if you change the color in the colorwheel. You can also change the values with the sliders.
3. A slider that controls the intensity of the color. This corresponds to the V in HSV.
4. A pipette capable of sampling colors from any Blender window or render window.
5. Buttons that can change it to "HSV" or "HEX" mode.
- Alternatively you can specify hue, saturation and value components by clicking LMB on the "HSV" button and pushing the sliders around accordingly.

- You can also press the last button and enter the hex-adecimal (or HEX) code. This is simply a different representation for RGB, where the hex digits represent *rrggb*.

HSV is probably the most easily understandable way of specifying and experimenting with colours. However, as is common with most computer systems, all colours in Blender are represented internally as RGB.

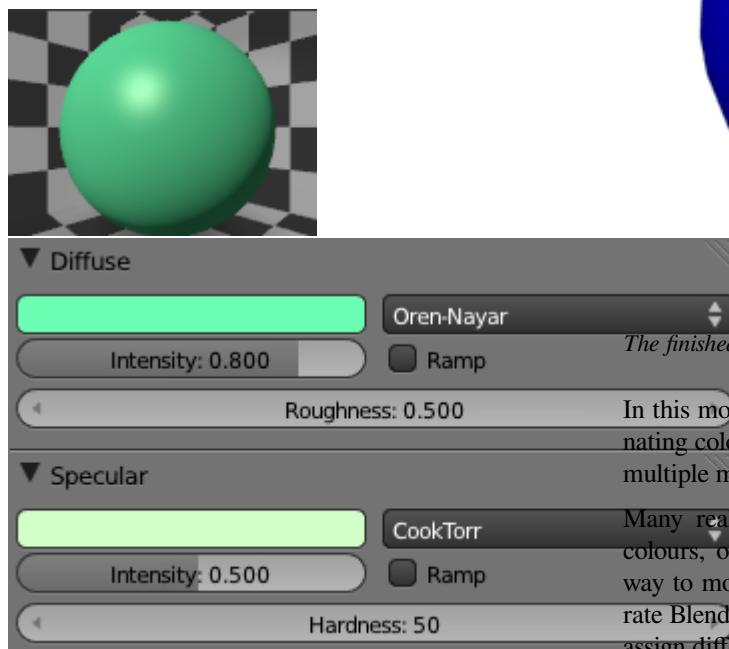
If you want to get rid of the window just click LMB anywhere else.

The most used method of creating a color of your own is using the color wheel, but because we want to be sure you will get the exact same color as us we will use the sliders. Use the above methods to set the diffuse color to R=0.149, G=1.000, B=0.446 (or use the HEX code: 6CFFB2). If you look in the “Preview” panel, you will see that the material is now bright green.

Most real-life materials (other than metals) don't alter the color of specular light. For this reason, Specular and Mirror are usually left at their default values (white). For green ooze, however, you'll disregard this rule-of-thumb:

1. Click LMB the sample rectangle below the Specular window.
2. Use the color selection dialog to adjust the specular color and watch the Preview panel to see how this color affects the sample sphere's highlight.
3. Set the specular color to R=0.640, G=0.990, B=0.566 (or use the HEX code: D1FEC6).

With these values for Color and Specular, you should be able to get a good ooze later on. The Preview, Diffuse and Specular panel should now look like this:



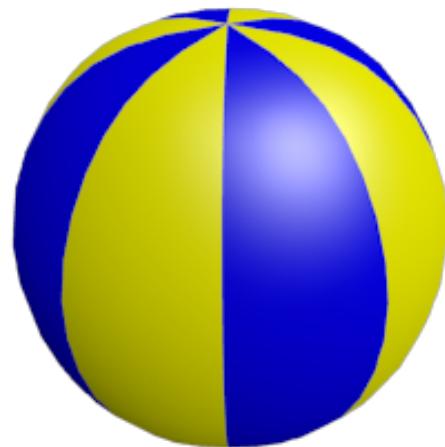
As you can see, there are many other material buttons. Many of these will be explained in later modules. Suggestions for creating specific materials may be found in the “Every Material Known to Man” module.

Save this scene before proceeding. You will need it for the “Quickie Texture” module, in which you will perfect your ooze.

2.16.3 Additional Resources

- [HSL and HSV](#) at Wikipedia.
- [RGB color model](#) at Wikipedia.
- How to Assign a Different Material to Each Face of a Cube at <http://www.youtube.com/watch?v=hCYViRJFf5w>

2.17 Multiple Materials per Object

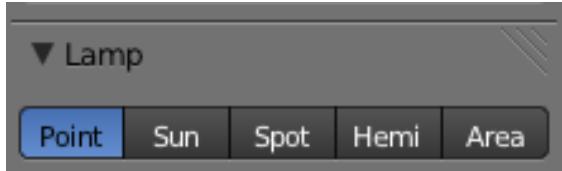


In this module, you'll create a beach ball with two alternating colours. Along the way, you'll learn how to apply multiple materials to a single object.

Many real-life objects have parts which are different colours, or are even made of different materials. One way to model such objects is to make each part a separate Blender object. However, Blender also allows you to assign different materials to parts of a single object.

2.17.1 Set the Scene

Begin by opening Blender and removing the default cube. Then select the lamp and change its type from 'Point' to 'Hemi' in the Light settings in the properties window (object data button), this has the advantage of it giving more light.



The options for changing lamp type in the properties window

Now create a mesh for the beach ball:

1. With the 3D View window active, press Shift + A) and choose Add → Mesh → UV Sphere.
2. In the "Add UV Sphere" panel in the bottom of the tool shelf, specify 8 segments and 4 rings.

The initial result will be crude, but meshes with fewer vertices are easier to edit.

Make the mesh rounder and more organic using automatic subdivision:

1. In the "Properties" editor, select the "Modifiers" context (wrench icon).
2. Select "Add Modifier" and click Generate → Subdivision Surface.
3. For the number of subdivisions, set both the 'View' and 'Render' count to 2.

Get rid of that blocky look:

1. Ensure you're in Object Mode.
2. In the tool shelf, select the "Tools" tab.
3. In the "Edit" panel of that tab, set the shading to *Smooth*.

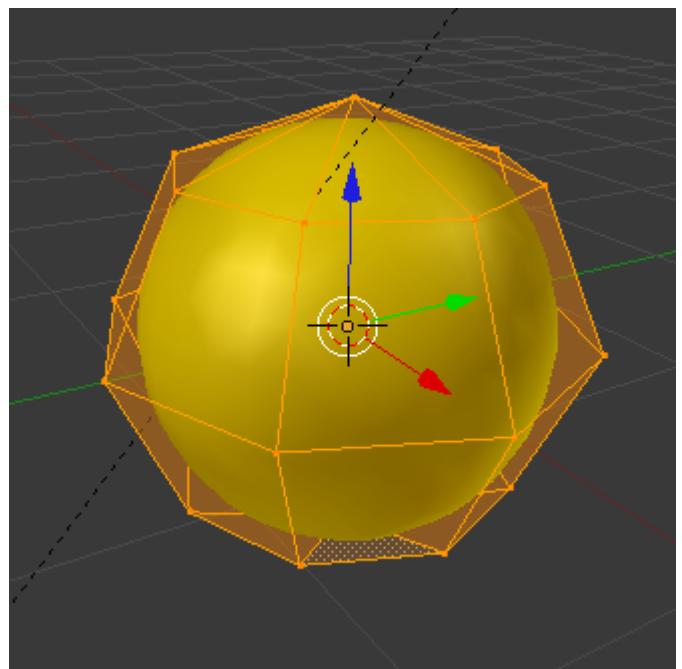
The ball is now round, but a bit prolate. To make it more spherical, scale it by about 1.1 along the X and Y axes. To select the X-Y plane, you select *'not Z'*, by using the key combination Shift + Z . The complete sequence is, then, S , Shift + Z , 1.1 .

2.17.2 Colorize Time

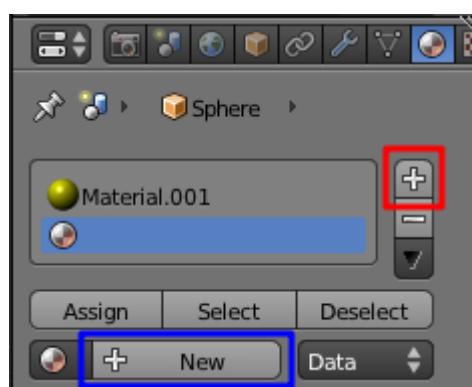
Now you're ready to begin adding colors to the object:

1. Press Tab to put Blender into Edit mode.
 2. In the Properties editor, select the "Material" button
 3. Press "+ New".
- A new material appears in the material slot list, and several additional panels appear below to edit the created material.
4. In the "Diffuse" panel, click on the default white diffuse color and change it to a nice yellow.

At this point, the entire ball is yellow.



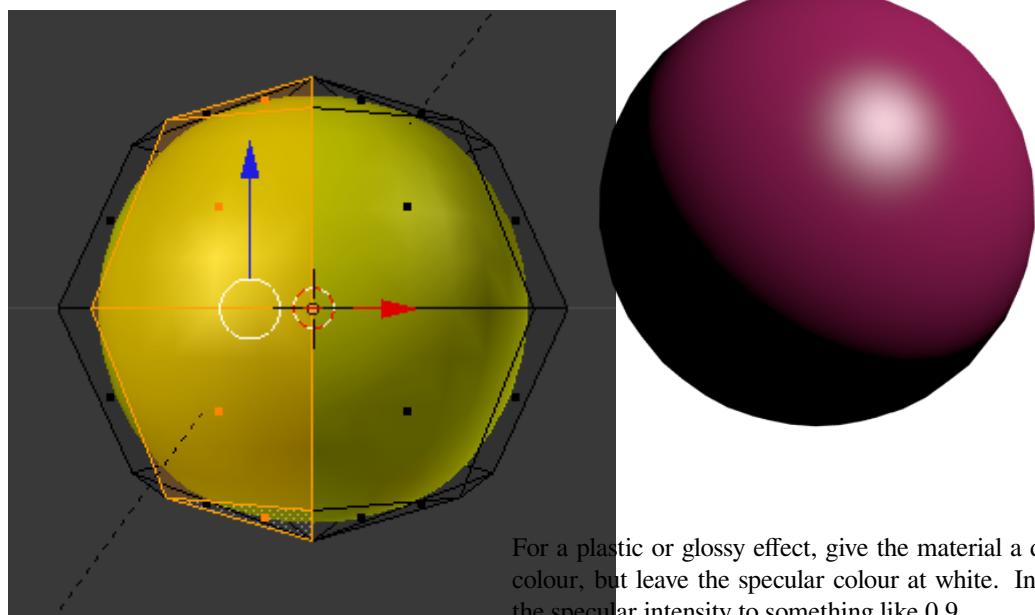
In the "Materials" panel click the "+" button (indicated by the red box in the picture, below) next to the material slot list to create a new blank slot. The "+ New" button will reappear (indicated by the blue box, in the picture below).



Click the "+ New" button and a new material will be created and assigned the empty slot in the materials slot list. Ensure that the new material is selected, then change the diffuse color to blue. Nothing will happen to the beach ball, yet.

Now make a single blue stripe on the ball:

1. All the vertices should still be selected from before; make sure the 3D view is active, then hit A to deselect them.
2. Switch to front view with NUM1, and to "Face Select" mode with Ctrl + Tab then F.
3. To avoid accidents, make sure that the "Limit selection to visible" option is enabled in the 3D View header bar.
4. Select a column of four faces that will make up one stripe of the beach ball (using SHIFT + RMB on each face):



5. In the "Material" property window, select the blue material slot in the list, then click the "Assign" button.

Rotate the view (e.g. NUM6) so you can skip past a yellow stripe adjacent to the blue stripe, and select the second column that will become a blue stripe. Work your way around the ball to do this three more times. (Remember we made the sphere with 8 segments; four of these are yellow, and four are blue).

Now you see the benefit of making a sphere with only 4 rings: more rings would have meant more faces in each stripe, and more clicking to select them.

2.18 Metal Versus Plastic

There are different kinds of shiny materials. Consider the difference between a shiny metallic object, and one made out of a glossy nonmetallic material (like plastic or ceramic). This page will explain some simple techniques for (approximately) mimicking the appearances of these materials using shader settings in the Blender Internal renderer.

For all the following manipulations, start a new Blender document, get rid of the default cube, and replace it with a UV sphere. Set it to be smooth-shaded. Change the lamp falloff to be inverse linear, just to make the scene brighter. Assign the sphere a new default material. It is the settings of this material we will now proceed to play around with.

2.18.1 Making It Plastic

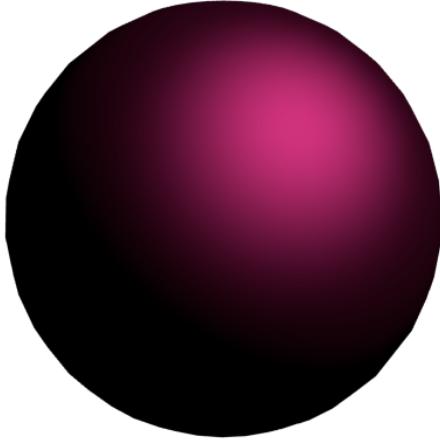
For a plastic or glossy effect, give the material a diffuse colour, but leave the specular colour at white. Increase the specular intensity to something like 0.9.

(In this and the following examples, I used a diffuse colour of #E7398B.)

2.18.2 Making It Metal

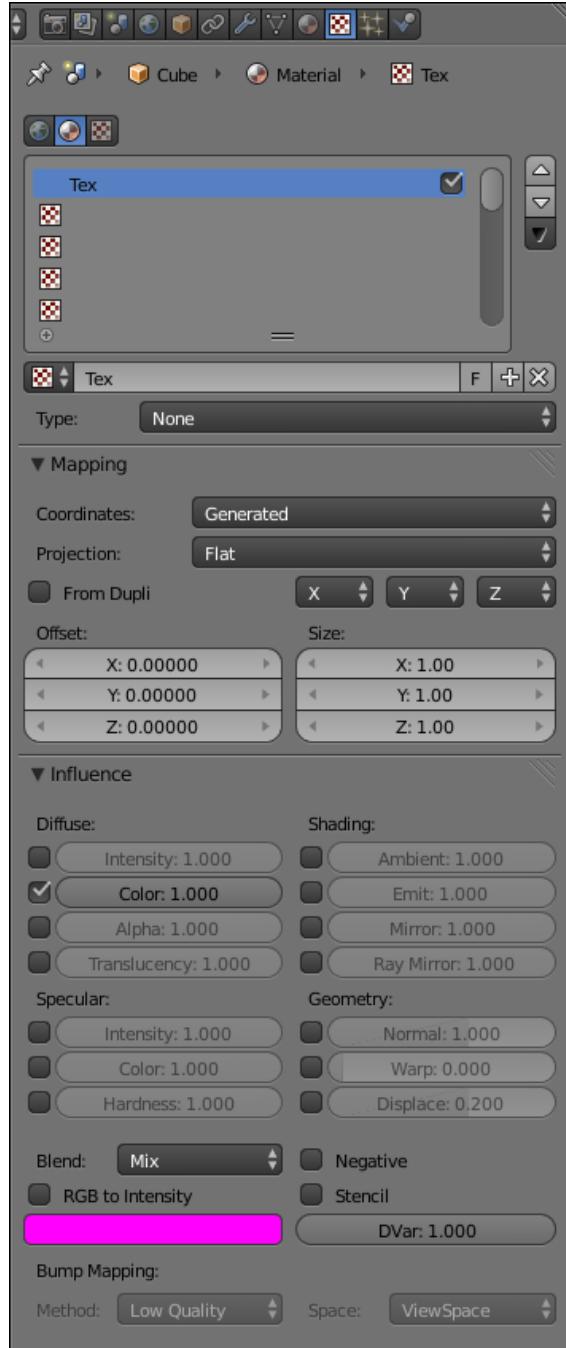
Now change the specular colour to be the same as the diffuse colour. (The easy way to do this is to bring up the specular colour picker, click on its eyedropper icon, and use the eyedropper tool to click on the swatch showing the diffuse colour.) Also lower the specular hardness from its default value of 50, to something like 25 or even 12. This will spread out the specular highlight, giving the impression of a surface that is shiny, but not perfectly smooth. Also lower the diffuse intensity, to something like 0.05.

2.19 Texture Settings



To make the metal more convincing, you may want to choose a more typical metal colour, like grey, copper or bronze.

2.18.3 Making It Ceramic



Material texture settings

Let's try for a glazed-ceramic look, or perhaps some dark, shiny stone like obsidian. Set the specular colour back to white (the easiest way to do this is to switch to HSV view in the colour picker and set the S(atisfaction) value to 0). Increase the hardness to something like 200 to narrow and intensify the specular highlight. Leave the specular intensity high and the diffuse intensity low.

To get an even more sharply-focused highlight, change the specular shader model from its “CookTorr” default to “WardIso”. The “Hardness” parameter gets replaced with a “Slope” instead; leave this at the default 0.1.

In the Properties window, you will find the Texture context, which looks like at right.

At the top you will see a row of three icons , which indicate texture settings to view and change:

- World Texture — a texture to use for the sky backdrop

-  Material Texture — a texture associated with the currently-selected material
-  Brush Texture — a texture used for some other purpose.

There are two main types of textures in Blender:

- Image/Movie textures
- Procedural textures (all other types in the Type menu).

An image/movie texture allows a two-dimensional image (which might be static or moving) to be wrapped around the surface of a three-dimensional object in some way. Alternatively, a procedural texture directly maps a pre-defined three-dimensional mathematical function to the surface coordinates of the object.

The “Coordinates:” popup menu defines how positions on the object surface are mapped to positions within the texture coordinate space. All of these options specify automatic mapping algorithms, except one: the “UV” option. This one lets you work within the UV/Image Editor, where you *unwrap* the surface of the mesh onto a flat rectangle showing the texture image (this really only works with Image/Movie textures), and then move sections of the mesh around to make them show corresponding parts of the texture.

The “Projection:” popup menu further controls how the two-dimensional surface of the object is mapped to a two-dimensional Image/Movie texture. It seems to have no effect for procedural textures.

The “Offset:” and “Size:” X, Y and Z values allow simple adjustments of the position and scaling of the texture. Note that the size values work the opposite way to what you might expect: larger values here make the texture *smaller* along the corresponding dimension.

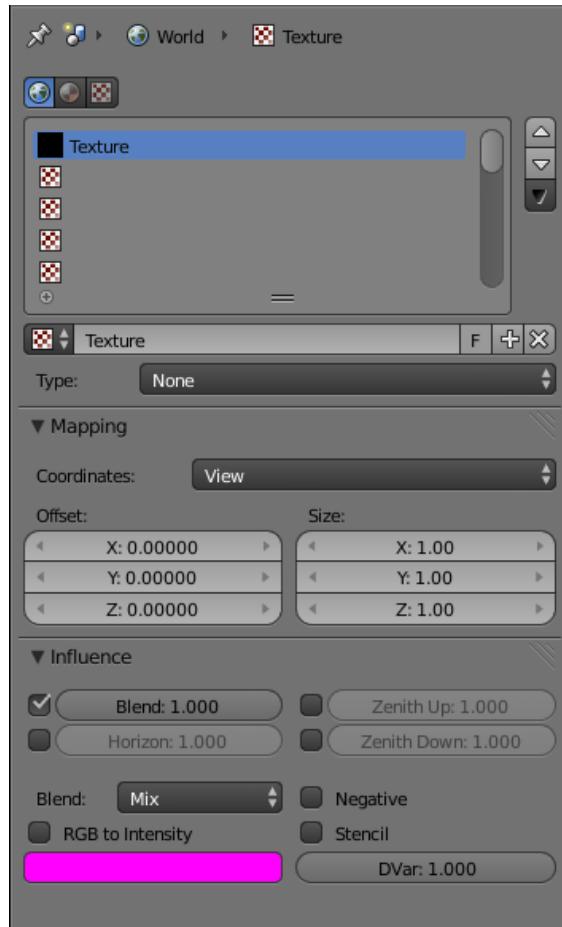
The image at right shows the common settings panels for all material texture types. Additional panels will appear depending on the chosen texture type; the settings shown are for the “None” texture, which is the same as having no texture at all.

2.19.1 Material Textures

A material may have more than one texture associated with it. At the top of the material texture settings (see above), is a list of the *texture slots* associated with the material. Slots may be empty (unused), and slots containing a texture may be enabled or disabled, by checking or unchecking the box at the right of the list item. Disabling a texture slot stops it having an effect on the material, which is the same as deleting the texture from the slot

altogether, except it is easier to revert. This can be useful when trying to debug the effect of a combination of textures on the material.

2.19.2 World Textures

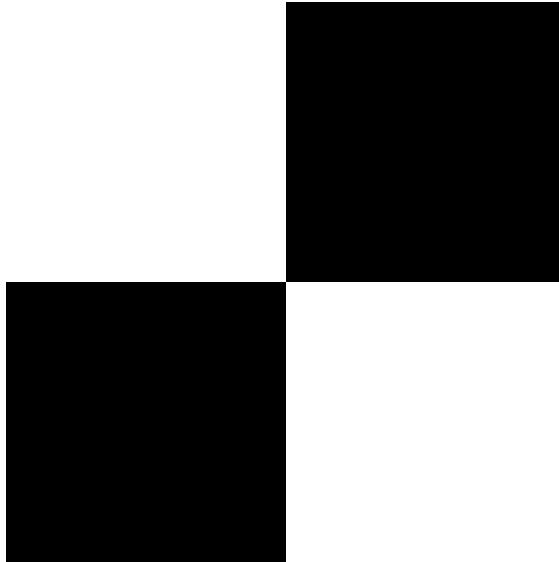


You previously saw how to set up colours for the sky in the **World Settings**; you can also add a sky texture as well.

World texture settings look similar to texture settings: again there are a number of slots, and there are mapping and influence options. But the mapping coordinates types are different (and there is no UV option), and the influence types are more limited.

If you’re wondering why your texture definition here is making no difference to your sky, either make sure the

 “Blend” checkbox is checked in your World  settings, or check the “Horizon” Influence box here if you don’t want a sky gradation.

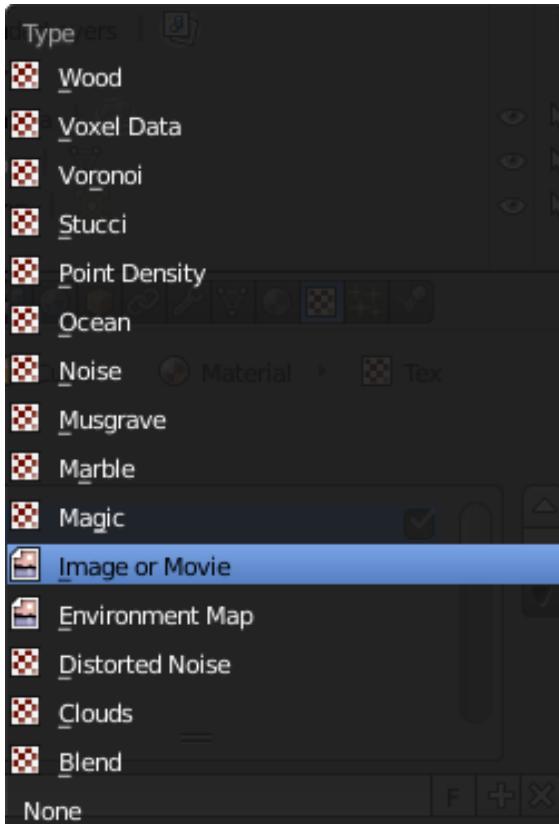


Simple checkerboard

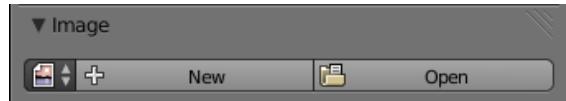
2.20 Image Textures

2.20.1 Image Texture Settings

To understand how the texture settings apply to image/movie textures, start with an example texture. A nice simple one is this checkerboard at right—don’t forget to download it in (or convert it to) PNG format, as Blender cannot use an SVG file as a texture.



Start a new Blender document. Note the default cube already has a default grey material, called “Material”, and this already has a single texture, called “Tex”, of type “None”, which means it has no effect.



Under the Texture context of the Properties window, with Material Texture selected, change the texture type to “Image or Movie”. You will immediately see some new panels pop up in the texture context. Look for the Image panel, as at right. This initially contains a popup menu icon for selecting from any previously-loaded images (this will start out empty), a “New” button for using one of Blender’s predefined test textures, and an “Open” button for loading an image from a file.

Click the “Open” button, and select your previously-downloaded or converted PNG version of the example checkerboard texture.

Now a whole lot more settings will become visible. From the top, the panels are:

- Preview — gives you a simple display of how the texture looks.
- Colors — lets you make simple adjustments to the image brightness, contrast etc.
- Image — lets you choose from any already-loaded images, and shows you the pathname of the file the image was loaded from. Note the two arrows in a circle to the right of the pathname display: clicking this will tell Blender to reload the image from the file, which is useful if you make changes to it in an external image editor.
- Image Sampling — controls how the image can be interpreted in a different way from straight pixel values.
- Image Mapping — lets you crop the input image, and apply fixed numbers of repetitions to it along each axis, even before it goes through the usual texture-tiling repetition process.
- Mapping, Influence — these are more general panels that apply to all types of textures. They will be discussed in more detail shortly.

If you render F12 now, you should end up with an image like this. Notice in the Projection menu (*Texture → Mapping*) the initial selection is “Flat”: this means that the texture X and Y coordinates go straight to object X and Y coordinates. Thus, the texture only appears on the top (and also bottom) of the cube, not on its sides. See

also the three little popup menus just below the Projection menu, each containing the items X, Y and Z. These let you rearrange the object coordinates that the texture coordinates map to. If you change the first two, you can get the texture to appear on other pairs of sides of the cube, other than the top and bottom. The third menu (corresponding to the Z axis of the texture) has no effect (yet), because a flat image texture is only two-dimensional.

Now try changing the three “Size” fields in the Mapping panel: give them all a value of 3. This will uniformly shrink the texture pattern to one-third of its original size. Or alternatively, it will require three times the number of texture repetitions to span the same distance as the original.

Now let's try the other Projection types. Here's what “Sphere” looks like. Imagine the texture pattern as a flat sheet stretched and curved around, and its edges joined to form a sphere surrounding the actual object; then the sphere is shrinkwrapped down onto the object.

Note the top and bottom edges of the sheet shrink down to single points at the north and south poles; this is why the squares of the checkerboard pattern turn into triangles next to these points.

Here's a Tube mapping. Here the texture pattern sheet is rolled round into a cylinder, with only one pair of edges joined together, the top and bottom left open.

And lastly, here is a Cube mapping. Here 6 copies of the texture pattern are arranged parallel to the faces of a cube, before being shrinkwrapped onto the actual object. Which in this case, happens to be a cube.

Cube mappings are very commonly used in game engines, because they are just about the simplest way to wrap a texture around an entire object.

2.20.2 Making Your Own Texture

Procedural texturing is very powerful; however, sometimes it is difficult or impossible to generate the desired realism with them. Image texturing is there for you when you need it. To review, the basic idea is to take an outside image and wrap it around your model. You can use any texture, or a seamless one if you want it to repeat to get a tiled effect. The following shows how you create a seamless texture, and then how to apply any texture (seamless or otherwise) to an object.

The difference between 'tiled' and 'seamless'

In many cases a simple material will just not cut it for an object, and you will want to apply a texture to it. However, depending on the object, you may want to apply either a seamless or tileable texture. A seamless texture is an image that will, when applied to an object, spread evenly across the surface of the object without any visible

borders or ‘seams’ even if the object is many times larger than the resolution of the image (also called ‘procedural textures’ in Blender). These can be useful in many situations; such as when you want a texture for a carpet to seamlessly repeat itself without having a huge resolution.

A tileable texture on the other hand, is an image that will repeat itself across an object, but with noticeable seams. Any image can be used as a tileable texture, but often they will only be used in specific instances such as a vinyl floor with a tiled pattern on it.

See [Using Textures](#) for more details on applying images as textures, and using them to affect many other surface attributes such as luminosity, reflectivity, translucency, displacement etc.

How to make a tileable texture with the GIMP

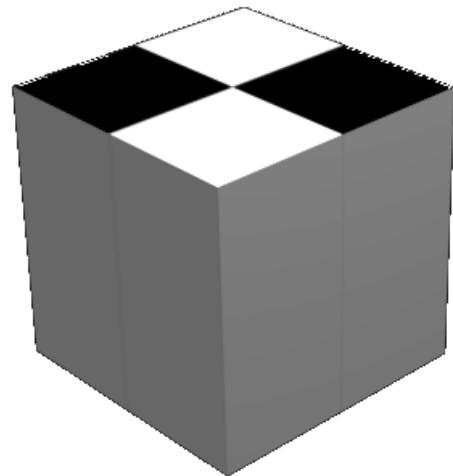
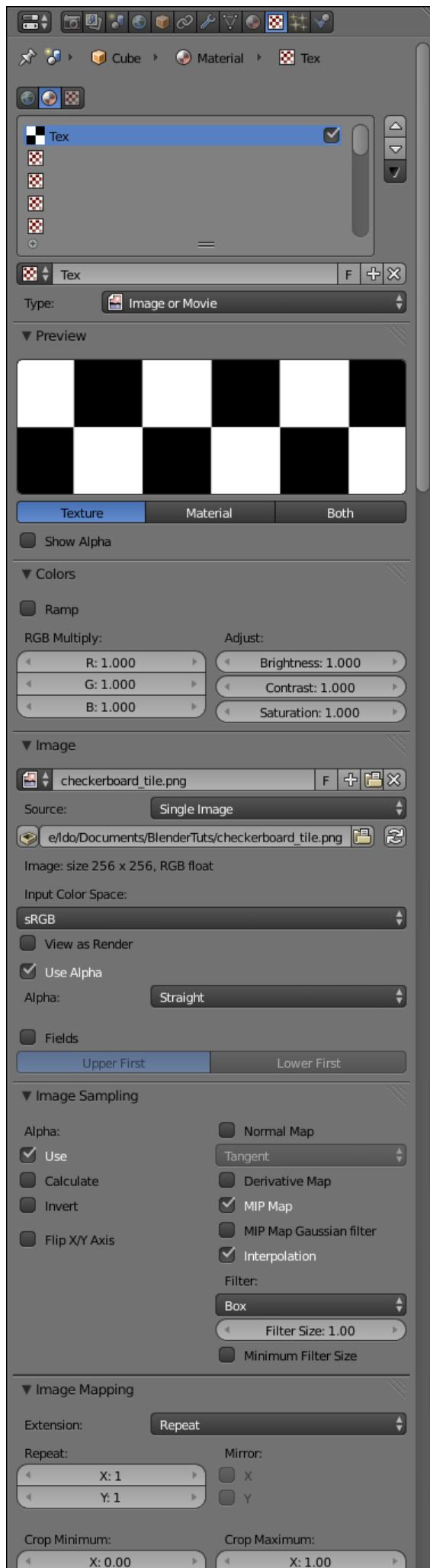
It is easy to create a tiling texture image with [the GIMP](#). Start with the photo you want to use. Crop out any part you don't want. Here's an example random photo of some plants in my garden:

Go to Gimp's “Filters” menu, and find the “Map” submenu. In here you will find the entry “Make Seamless”. Select it. That's it:

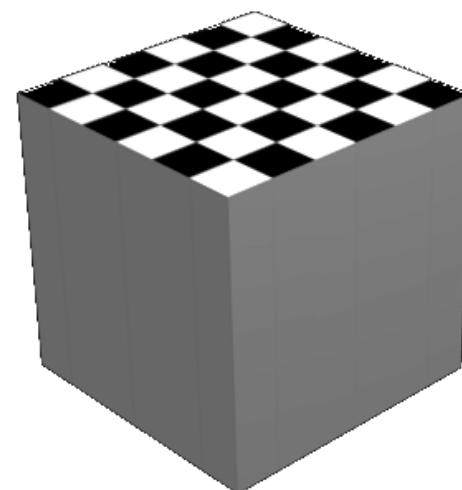
Just to prove it works, here's a (scaled-down) use of the result as a tiled fill pattern:

Other Image Texture Editors

- [Wood Workshop](#) A free utility (Requires Operating System: Windows 2000/XP) that generates surprisingly high quality tiling wood texture images. These textures can be exported as standard image files for use within Blender.
- [MapZone](#) A free utility for Windows (works perfectly in Wine) that generates node based procedural texture maps. Mapzone can export diffuse, normal and alpha texture maps as standard image files. It can also import SVG regions created with Blender's UV mapping tools.



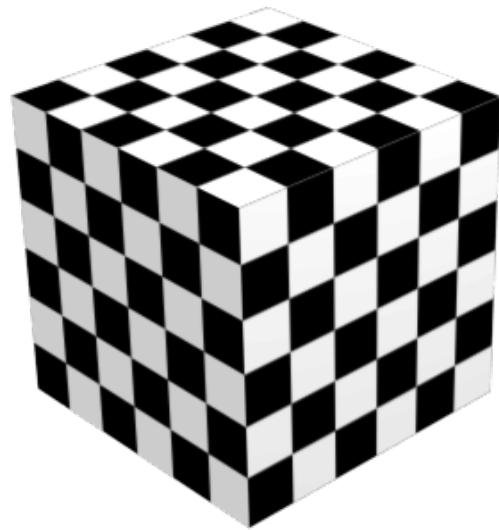
Projection: flat; axes: X→X, Y→Y, Z→Z



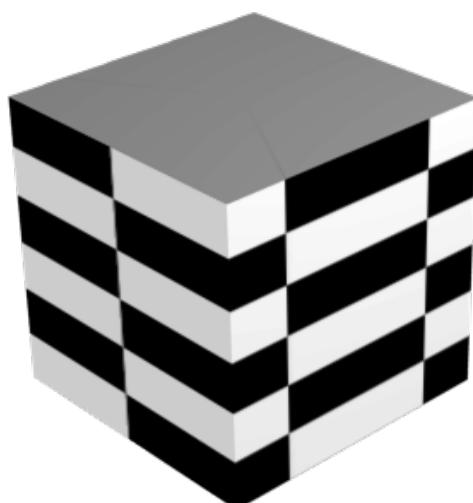
Texture size increased to 3.



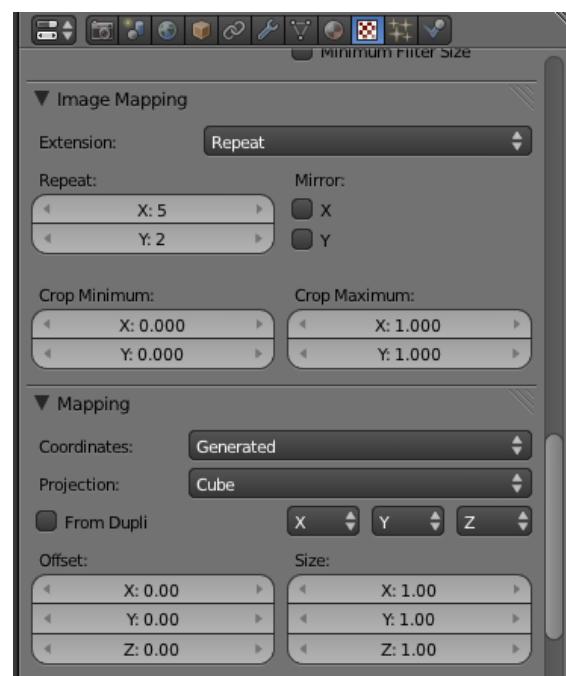
Texture projection set to Sphere



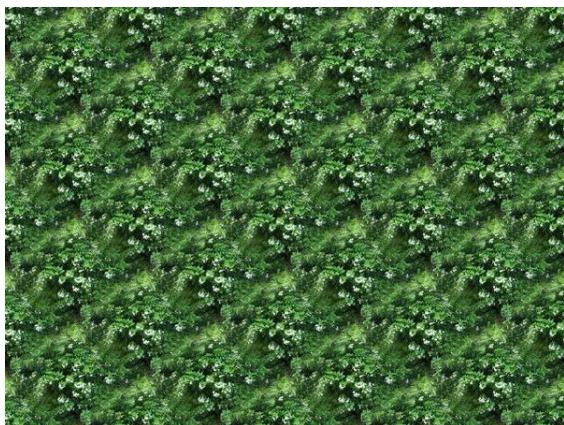
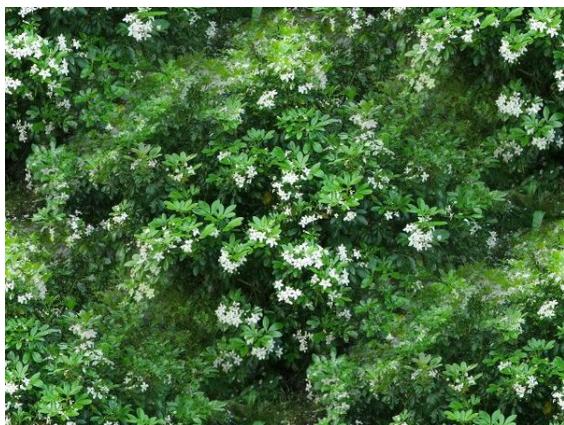
Texture projection set to Cube



Texture projection set to Tube



"Mapping" versus "Image Mapping"



2.21 Procedural Textures

Procedural Textures

Texturing objects can be broken down into two categories: procedural and image texturing. Procedural texturing makes use of mathematical formulas to generate textures. This is nice because it can be used to make relatively nice looking textures without external images which are very temperamental where you put them. Procedural Textures are all stored in the .blend file. These textures are obviously generated within Blender itself. Image texturing uses images created or captured outside of Blender, either from an image manipulation program such as the Paint.NET, GIMP or Photoshop, or captured on a camera. We have already learned about image texturing, so let's move on to procedural texturing.

Current Procedural Textures

Blender currently supports many procedural textures, including: Clouds, Marble, Stucci, Wood, Magic, Blend, Noise, Musgrave, Voronoi and DistortedNoise.

2.21.1 A Simple Wood Texture

Let's define a simple wood texture:

- Start a new Blender document containing the default cube.
- Select the cube (and nothing else).
- In the Properties window, go to the World tab  and turn on Environment Lighting (you can leave its default energy at 1.0).
- Go to the Materials tab , and rename the default "Material" to "Wood Material". Alternatively, delete the default material using the X to the right of the name field and add a new material.

Let's add some color and texture. You can see the results at any time by pressing F12 to re-render the scene.

Start by painting the cube a base color using the Wood Material's "diffuse" color:

- In the "Material" tab,
- Scroll down to the "Diffuse" properties panel and choose a darker brown color e.g. #A57E3F.

See http://en.wikipedia.org/wiki/HSL_and_HSV for where brown fits in the color wheel.

Next, let's add a texture to give the material some highlights.

- Switch to the "Texture" properties tab , and again rename the default "Tex" to "Wood Texture" or create a new texture. Notice at the very top of the "Texture" tab "Cube > Wood Material > Wood Texture"
- Change the Type of the material to "Wood" using the pop-up menu.

The texture sample will show parallel alternating black and white bars that don't look very woody at all. Never fear! The black regions will be the material's base "diffuse" color. The white regions are like "highlights" that will be painted over the base.

Let's make some improvements to the texture:

- While still in the "Textures" tab,
- Scroll to the "Wood" properties panel that appears, change the waveform from "Sine" to "Saw".
- In the next row of buttons down, change the type from the default "Bands" to "Ring Noise".
- Increase the Noise Size to 1.0.

Now the texture sample should show something resembling wavy tree-rings. If you hit F12 to render now, you will see these rings covering your cube, except a) the colour is wrong, and b) normal wood patterns aren't so nearly circular.

To make the pattern more elongated:

- Scroll to the "Mapping" properties panel,
- Change the Size X value to 2.0 and Y to 0.4. This squishes the pattern down along the X-axis, and stretches it out along the Y-axis, giving the elliptical tree-ring shapes you commonly see on wood planks and boards.

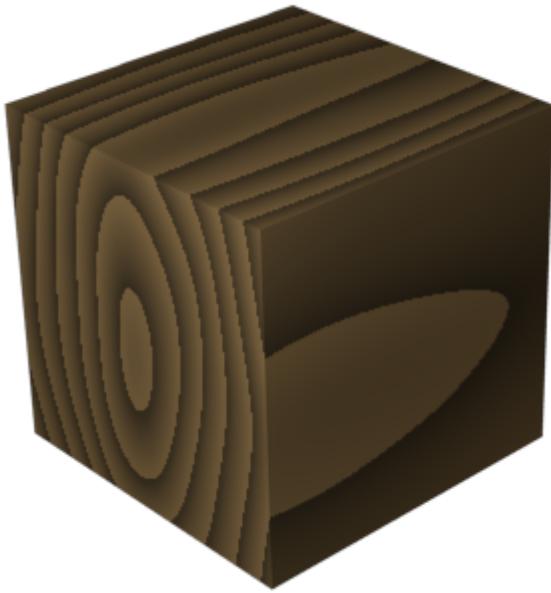
Hit F12 to render again, and the shape of the texture should be looking a lot more woody now.

The final step is to color the highlights in the texture:

- In the "Textures" tab,
- Scroll to the "Influence" properties panel further down,
- Click on the color swatch, and choose a nice brown colour.

For a nicer effect, I chose a very light brown e.g. #DEB887.

The result should look very woody indeed!



- Remember that you need to Render to see the wood grain on your object.

2.22 Quickie Texture

Textures are laid on top of materials to give them complicated colors and other effects. An object is covered with a material, which might contain several textures: An image texture of stone, a texture to make the stone look bumpy, and a texture to make the stone deform in different ways.

A texture may be an image or a computed function. What the texture does and how it is mapped onto your object is set in the material buttons. Some commonly used texture types are shown on the page [Using Textures](#).

This tutorial uses the file from the [Quickie Material](#) tutorial. If you didn't do it before, go back and do it now.

2.22.1 Making It Mottled

Step 1: Adding Texture to the Material

- In a Properties window, switch to **Texture** context.
- A default texture, **Tex**, should already be available and set to **Type: None**.
- If not, click one of the **Texture Slots** (the ones with chequered icons) and click the **New** button.

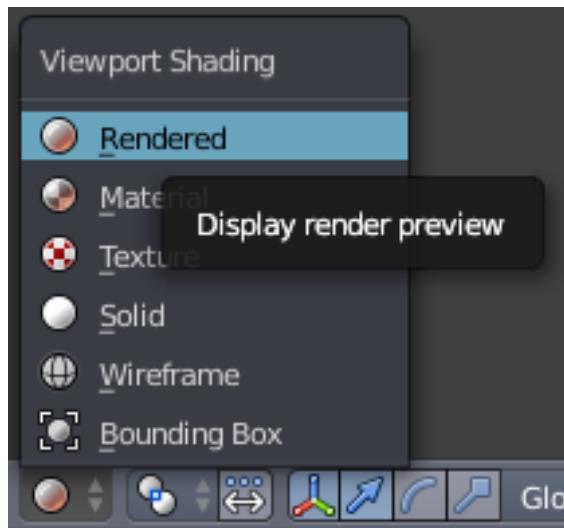


Texture Context with all the relevant panels.

- Set the **Type** to **Clouds**.
- The Texture **Preview** panel will now reflect this change. However, said change will not be reflected in the 3D view window.
 - You can do a quick render (**F12**) to see the change. However, you'll have to re-render every time you change a setting to see its effect.
 - Otherwise you can click the **Material** button in the Texture **Preview** panel to see the

changes to the material. (Click **Both** to see them side-by-side.)

- A better, albeit more resource intensive option would be to change the **Display Mode** to *Rendered*. (**Shift+Z** in the 3D view window or Selecting the Display mode from the 3D view Header.



Viewport Shading menu highlighting the **Rendered** option.

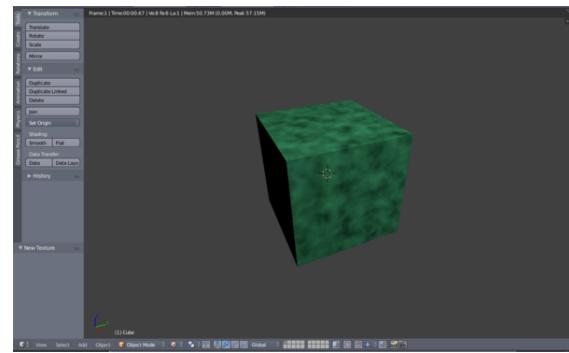
Step 2: Refining the Texture

- Once you use one of the ways to preview your work, you'll see Green and Magenta mixed in resembling a polished granite texture.
 - This is the default colour for any generated texture. Now all you have to do is change it to **black**.
 - But before that scroll down to the **Mapping** panel and make sure that *Coordinates* is set to **Generated, Global** or **Object** (for best results).
 - Scroll down to the **Influence** panel, and click on the colour swatch and drag the reticule in the bar to the right all the way down.
- Now the texture should look more or less like green granite

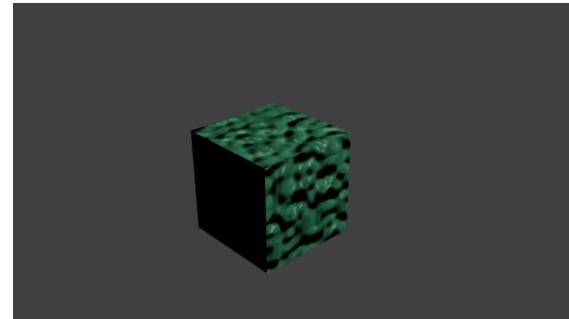
2.22.2 Making It Bumpy

Step 1: Adding a second Texture to the Material

- In a Properties window, switch to **Texture** context.
- The **Cloud** texture you just created will be listed in a slot.



Render result in 3D view.



Final render

- To create an additional texture click a second texture slot and then click **New** button.
 - Change the texture *Type* to **Stucci**.
 - Now if you preview this texture you'll only notice a bit of magenta mixed in with the previous texture.
- ### Step 2: Making the texture a Bump-Map
- Scroll down to the **Mapping** panel and make sure the *Coordinates* is set to **Generated, Global** or **Object** for best results.
 - Scroll down to **Influence** panel uncheck *Color* and check *Normal* under **Geometry**, then set it to **4**.
 - If required, set the *Method* under Bump Mapping to a higher Quality.

The render result should look like the one on the right.

Now mess around with the various settings we discussed, Particularly the settings in **Clouds/Stucci**, **Mapping** and **Influence** panels. Also try the whole tutorial (**Quickie Material & Quickie Texture**) with a sphere and other shapes.

2.22.3 Some Closing words

The downside of bump-mapping, as you may have noticed, is that it only provides an illusion of

depth/bumpiness. The edges will still be straight as in the render. For curved surfaces the outline will still look spotless while the centre looks deformed, plus shadows will still render smooth compromising the illusion. An alternative technique is displacement-mapping which actually deforms the mesh as per a texture to produce depth in the mesh, with the downside of creating a higher poly mesh.

With bump-mapping in general, you will get a greater effect on smoothly curved surfaces with high specularity as compared to flat surfaces with low specularity.

2.23 Halo Materials

2.23.1 Introduction

Halos are a neat effect. Instead of giving a colour/texture to the faces of a mesh, like normal Surface materials do, the Halo material ignores the faces and renders representations of the vertices instead. This can produce all kinds of ethereal, even ghostly, fantasy effects, of objects that look like they're made out of light rather than ordinary solid matter.

A halo material can also produce a *flare* effect. This is the "lens flare" that happens when a physical camera is aimed at a very bright light source; the spillage of light bouncing around inside the optics produces coloured rings and other interesting artifacts on top of the image. This has become such an accepted part of photography that computer graphics programs like Blender, which do not suffer the imperfections of physical lenses, go to a great deal of trouble to offer a realistic flare effect.

Flare effects can also be achieved using compositing node and a material with an "emit" value, such flares may in some circumstances render faster and be simpler to control. This works for the Blender internal render engine, as do flares generated with halos. This is done by opening the node editor (switching the 3d viewer tab to one of these for example) then clicking "compositing nodes" and "use nodes", "filters" can then be added to produce these effects.

This tutorial will show you how to create an image representing a flare effect in a picture of the Sun.

2.23.2 Setting The Scene

Open a new default Blender document. Get rid of the default cube. Insert a new UV Sphere mesh in its place, and set the number of segments and rings to 24 each. Also set Smooth shading. This will be your Sun. Create a new material for it, set the Diffuse colour to a suitable yellow. Under the Shading panel in the material settings, look for the "Emit:" slider and give it a value of 1.0 to make it look bright. Since the Sun emits its own light, you don't need

the separate default light, so get rid of that.



Go to the World properties tab . In the "World" sub-header, click on the colour swatch labelled "Horizon Color" and assign a nice deep blue colour for your sky.

If you do a render now, you should see your bright yellow orb, but without any flare effect.

2.23.3 Adding The Flare

Now add a new Circle mesh; the default 32 vertices should be enough. By default it lies in the X-Y plane, which again is fine. Move it along the Y-axis a little closer to the camera (negative-Y direction), until it lies outside your Sun sphere, but still close to it. Scale its size down by 0.5. (It will probably be invisible when first created, because it is initially inside your Sun sphere, but it will be initially selected, so you can immediately press G Y and start moving the mouse without pressing any buttons, and make it appear from inside the Sun). Create a new material for it, and set the type to Halo.

In the Halo panel in the Material settings, increase the size to 3.0—this is the size of the fuzzy image that is rendered around each vertex, and this value is sufficient for them all to run together into a continuous ring. Reduce the Alpha to 0.05 to avoid overpowering the image with the halo effect.

Go further down the halo Material settings, and find the Flare panel (in Blender 2.75 you can check "Flare" but what settings you do, nothing will work). Check the title box to enable this. Set the number of Subflares to, say, 8 (this controls the number of separate halo reflections that will be generated, though you probably won't be able to distinguish that many). Set the Boost to 10 to make the subhalos brighter than the original parent halo.

The Seed value in the Flare panel controls the particular flare pattern that you see; each number produces a different effect. I chose the value 3 for this example.

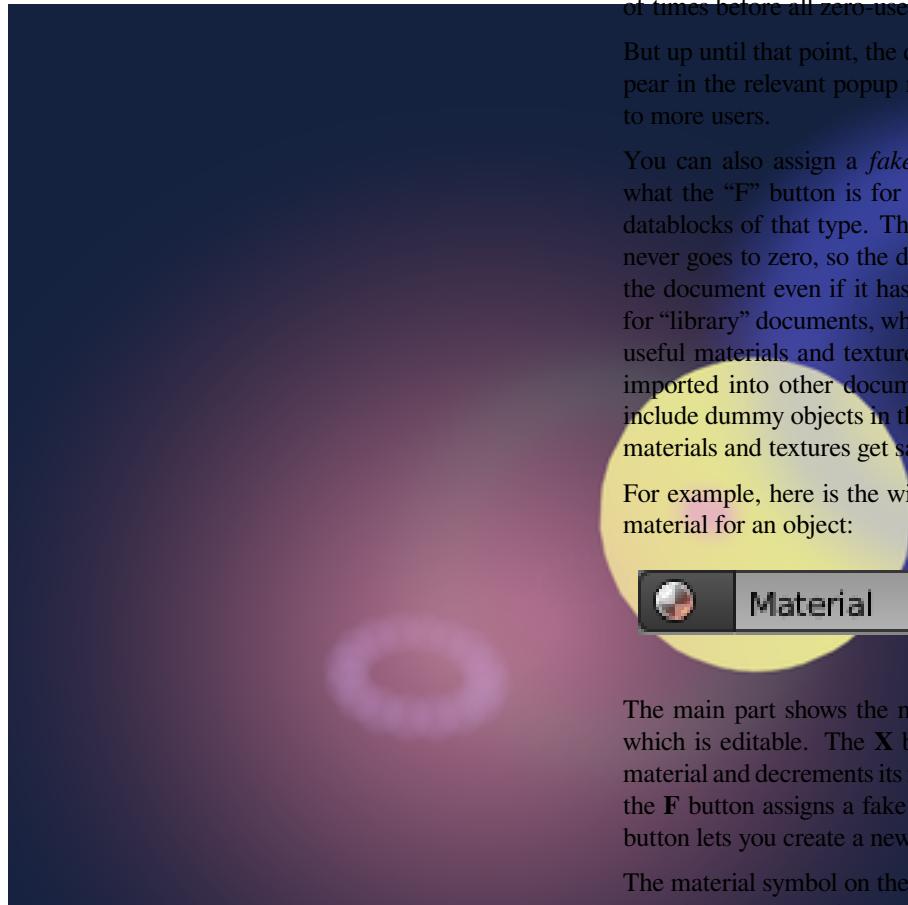
Where did the circle go? Like any object with a halo material, the circle object can be quite hard to see when it's not selected. If you lose track of it, there are a couple of ways to find it again:

- Select everything with A . Now you can look for the ring of dots and RMB on it to select it exclusively.
- Use the outliner window at the upper right. You should see it listed here under its default name of "Circle"; click with LMB to select it, and you should see the ring of dots appear in the 3D view.

If the circle object is still inside the Sun, then wireframe Z or bounding-box view modes may be helpful to find it again.

2.23.4 The Final Result

Now hit F12 to render, and you should see something like this (the flare effect may not appear immediately with the rest of the image, give it a few more seconds to appear):



Exercises: Try different positions for the circle mesh; move it near to the Sun (even partly in it), far from it, move it around to different sides. How does this affect the flare pattern? Also try changing the size of the circle mesh.

2.24 Blender Memory Management

2.24.1 Datablocks And Users

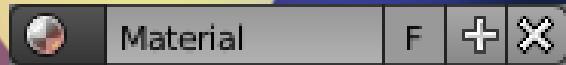
It is helpful to understand how Blender manages memory. Just about everything in a Blender document—objects in scenes, scenes themselves, materials, textures, whatever—is stored in a *datablock*. Each datablock has a name, which must be unique among datablocks of the same type. Each datablock may be referenced from one or more places, mostly in other datablocks—in Blender parlance, it has one or more *users*. For example, several different objects might share the same material, so when you change the characteristics of the material, it automatically changes the appearance of all those objects.

If the number of users of a datablock drops to zero, it still stays around in memory, but *it will not be saved when the document is saved*. Thus, if you save and reload the document, all the datablocks with zero users will disappear. (In some cases you may need to save and reload a couple of times before all zero-user datablocks disappear.)

But up until that point, the datablock will continue to appear in the relevant popup menus, so you can reassign it to more users.

You can also assign a *fake user* to a datablock; this is what the “F” button is for in the popup menus that list datablocks of that type. This ensures that the user count never goes to zero, so the datablock always gets saved in the document even if it has no real users. This is useful for “library” documents, which can contain collections of useful materials and textures, say, that can be linked or imported into other documents, without also having to include dummy objects in the library just to ensure those materials and textures get saved.

For example, here is the widget that lets you choose the material for an object:



The main part shows the name of the current material, which is editable. The X button breaks the link to this material and decrements its number of users by one, while the F button assigns a fake user to this material. The + button lets you create a new material.

The material symbol on the left pops up a list of existing materials to choose from, plus a search box to search all existing materials:

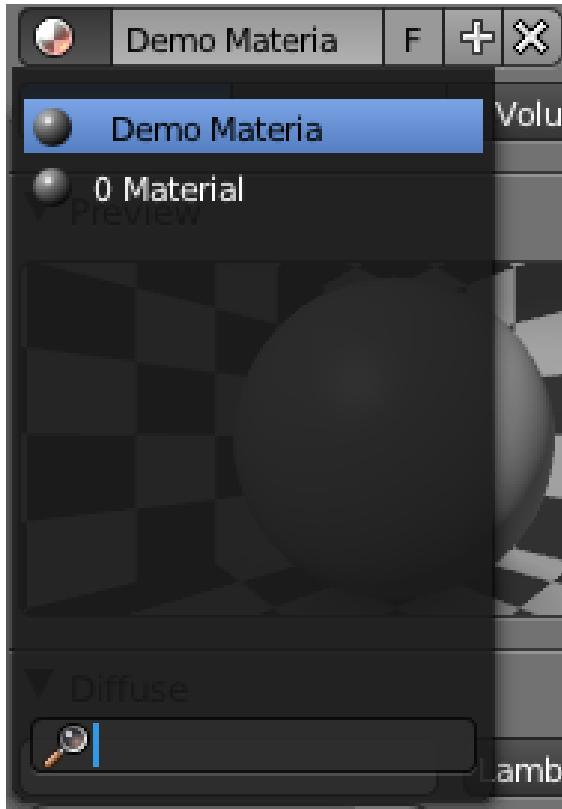
Note the entry with the 0 symbol next to it; that currently has a user count of zero, and will disappear when the document is saved and reloaded, if it is not further used.

The widget also displays the current user count if it is greater than 1:

In this case the count was incremented because the F button was selected.

This is the basis of the (slightly confusing) distinction in Blender between *object datablocks* and *object data datablocks*. Object datablocks contain the information common to all the *types* of objects in the 3D scene, regardless of whether they’re mesh objects, lamp objects, camera objects or whatever; whereas the *object data* datablocks contain the information specific to that *instance* of the type of object, e.g. the vertex, edge and face definitions for this particular mesh you might be using, or the colour and energy of a lamp you’ve set up for your project, or the field of view of a camera you have.

Which leads us to the difference between the two object duplication commands, SHIFT + D and ALT + D : the former duplicates both the object datablocks and the ob-



ject *data* datablocks (though this can be controlled in your User Preferences), while the latter only duplicates the object datablock. What that means is that in the first case the two objects are truly independent, but in the second case the new object continues to share the same object *data* datablock so a change in one will result in a change in both of them. So, for instance, if you use ALT + D on a mesh object and edit the vertices, edges or faces on one copy, the other copy will also be affected.

2.25 Using Bones

Bones are used for shifting models and making them posable. If you are not ready for this yet and wish to continue simply modeling, please skip this tutorial to the next section.

Bones are a modeling tool that are especially important for animating characters. Bones allow you to move characters' limbs in a way that is much simpler than trying to re-arrange the vertices every time.

It works by associating a bone with particular vertices, causing them to move along with the bone when the position is changed in pose mode. Using bones is fairly simple once you get the hang of it, but, like many things in

Blender, can be a little daunting at first sight.

Bones don't do much on their own; in fact, they turn invisible at render time. For this following module we'll use the character that we had made by the end of the module **Putting Hat on Person**. You will have to have completed all the modules in Section 2B. Note that while we will be using bones on a simple person, the process can be used with any creature or body type you imagine!

2.25.1 Laying down bones



Note: This just shows the basics of adding bones to an object. Go to the advanced animation page for a more comprehensive guide on this.

First of all, we'll need a model to put some bones on! For this tutorial, we're going to use a humanoid model. Open the model that you had created by the end of the **Putting Hat on Person** tutorial, or download a pre-made model from [here](#).

Here's our setup, with Block Dude standing on a plane. You can add a plane by pressing Shift + A → Mesh → Plane. Scale the plane to an appropriate size and move it so that it is approximately underneath the person.

Noob note: You will be placing armatures ("bones") inside your humanoid, so you must work in "wireframe mode", not "solid mode". Otherwise, you will not be able to see the armatures when you place them. To toggle between "solid" and "wireframe", press Z. You may find it helpful to make the wireframe less complex by hiding the subsurface mesh. You can do this by going to the Mod-

ifier context panel of the Properties window and deselecting the eye button.

Note: An alternative to working in "wireframe mode" is to turn "X-Ray" on for the armature. To do this, select the armature. In the properties panel under *object data* there is a display menu. Click "X-Ray" in the second field of buttons. This will allow the armature to show through other objects.

2.25.2 Add a bone

Now, let's put some bones on Block Dude! In Object Mode press Shift + A → Armature → Single Bone.



What we are looking at is an armature. This is a single bone. Now, we need to put the bone in Block Dude! Move and rotate the bone so that it's in the middle of Block Dude's chest. If your bone does not have the correct length, then change the size of the bone by moving one of the ends of the bone: switch to Edit Mode, select one of the ends of the bone, then move it using G . Alternatively, you can scale it using S

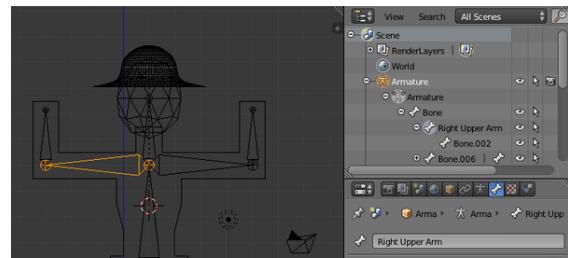
2.25.3 Extrude a second Bone



To create a second bone starting from one of the ends of the first bone, switch to Edit Mode with the bone selected, select the end of the bone, then extrude E the end. A second bone appears, with its start point on the selected end of the first bone. Move the mouse to position the end point, then press LMB , Enter , or Space . Scale the bone as needed to fit it in his body, and continue adding bones by extruding the end points. These operate much the same way as vertices: you can extrude, rotate, move, and even subdivide. Your finished result should look something like this:

2.25.4 Name the bones

Now, just to make things easier, we're going to name the bones. For example, my bones are named "Right Forearm", "Left Forearm", "Right Upper Arm", etc. While in Edit mode, select the bone you want to rename. In the Outliner , the bone you have selected will be visible with a circle around it . You may need to expand the Armature Object , Armature Data Object and



any parent bones before being able to view the selected bone. In the Bone context panel of the Properties window click the name field to edit the name.

Noob Note: When you are naming the bones remember that if you are looking at the person from the front, your left is the person's right. To make the naming easier switch to viewing the person from behind using Ctrl + Num1 .)

2.25.5 Parent the bones

Now, we need to parent the bones to the mesh. Go back into Object Mode and select Block Dude (and the Hat, assuming you made one). Now, select the Armature as well, so that it is the last object selected, and press Ctrl + P . The Parenting Menu will pop up. Select *Armature Deform → With Automatic Weights*. The person (and hat) are now children of the armature.

Noob Note: The selection order is important in defining which object is the parent, so you cannot select both objects at the same time. You must select the armature last to make it the parent.

2.25.6 Moving the Bones



To move individual bones, you have to go into Pose Mode. Select the Armature in Object Mode and switch to Pose Mode by pressing Ctrl + Tab or selecting the mode in the mode selection menu of the 3D Viewer. Try moving a bone around by pressing RMB to select it, and then hitting G or R to move it.

If you've done everything correctly, your mesh should

move when you move the bones! If this doesn't happen, scale the bones up so that they fit better in the mesh, and scale up the bones until they do what you want (read comment in the parenting section above on adjusting the bones envelopes if you do not get an effect while moving/rotating the bones). With the bones now, you can put Block Dude into a lot of different positions without moving individual vertices.

To the right is an example of how you can move Block Dude with the bones.

Also while in pose mode if after a RMB click you can't move bones with G or R , check the "Move Object Centers Only" button (just to the right of the Rotation/Scaling Pivot button).

2.25.7 In-Depth Info on Selected Bone Topics

Add/remove mesh from bone control



Noob Note: If you've been adding bones to your simple person from the previous lessons, you will have likely noticed that the hat seems to stretch when you move the arms in pose mode. To fix this, you will need to remove the hat from the forearm vertex groups created in the Parenting step.

To manually change the mesh areas that the bones control, go to Object Mode and select the object you want to add/remove (if the mesh is inside the same object, then select only the areas of the mesh you want to work with in Edit Mode).

In this case, select the Hat.

Switch to the Object Data context panel in the Properties window and scroll to the "Vertex Groups" submenu.

Now pick the bone group from the dropdown above the Assign/Remove buttons, and then hit Assign (or Remove) as necessary. Usually vertices will be assigned to one group, but can be assigned to multiple groups. In this case, we want to remove the Hat from the Forearm vertex groups. Select the Forearm vertex groups and press the remove button, as pictured. With both of the Forearm vertex groups removed from the hat, it should be able to move properly with the rest of the armature.

Mesh deforms like it's far away from the bones If the mesh is properly assigned to the bones they will move regardless of whether the bones are inside the volume of the mesh or not (HOW they deform WILL be affected however). The most common mistake in this step is creating and (more importantly) parenting the mesh to the armature while the armature is outside the mesh, which causes Blender not to assign vertices to any bone groups at all.

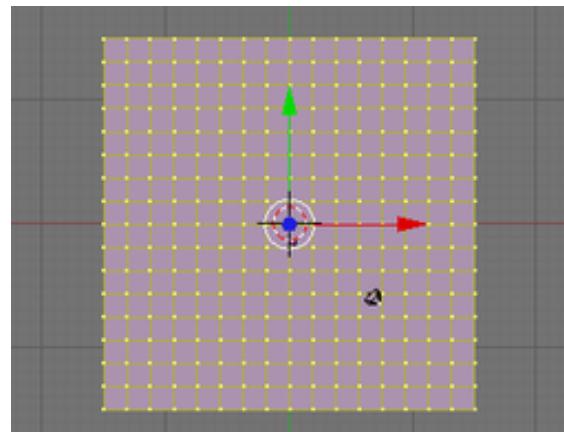
You can check this by editing the object (i.e. select the mesh and switch to Edit Mode, then un-select all vertices by pressing A until nothing is selected). Pick the *Object data context* then select a vertex group in the *Object data tab*, press **Select**. This will select the vertices associated with the bone group. If the wrong vertices appear selected, you need to assign them manually as explained above.

If there is no effect, in Edit mode select that bone (or bones) and choose Envelope display mode (**Properties** window), *Armature context panel → Display → Envelope*), then press Ctrl + Alt + S and increase its area of influence to cover all faces that should be influenced by the bone.

2.26 Mountains out of Molehills

Now that we've created our simple person, it's time to give him somewhere to go. In this tutorial we'll create a mountain range using a few simple, and handy tools.

2.26.1 Creating a simple plane



First we need a clean area to work with.

- Start off with a new project, using *File → New*, or hit *Ctrl + N* . If you have a default cube or plane just delete them now (select them with RMB and press X).

Our first step is to create a large grid plane that we'll use for the ground and grow our mountains out of.

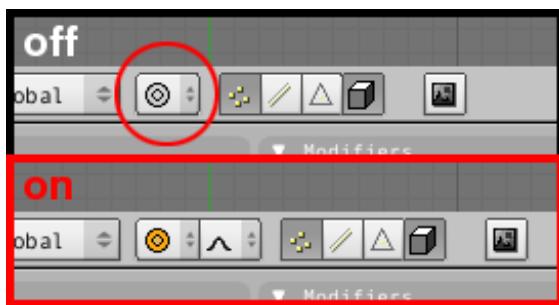
- Press NUM7 to enter top view. This way our grid plane will be lying flat when we create it.
- Press Shift + C . This sets the 3D cursor to (0,0,0) which will be the center of the grid we will add (or use - Shift + S → *Cursor to Center*).
- Now add the grid with Shift + A → *Mesh* → *Grid*. This will be our canvas.
- Now add more vertices to the grid. In the bottom of the toolbox window, change the number of X and Y subdivisions somewhere from 15 to 20.
- Change to Edit Mode using Tab
- Scale the grid plane up by about 15

First put the mouse close to the center of the grid plane and press S and drag the cursor away and watch the numbers in the bottom left of the 3D View. Hold Ctrl while dragging to increment by 0.1 for a more precise measurement. Alternatively, to enter the exact amount yourself, press S , then simply type **15** and hit Enter

2.26.2 First mountain

Now that we have the ground, it's time to start growing our mountains.

- Make sure you have nothing selected A .
- Select a random vertex with RMB . I usually start at the one that is 4 down from the top and 4 in from the left (the 4th vertex if you count the edges).
- Change to the side view with Num3 .



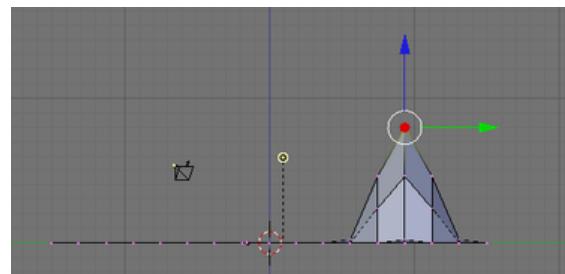
- Press O to change to proportional edit mode or use the button which shows a grey ring on the header of the 3D View. The button will change its color to blue. You can also use Space → *Transform*→*Proportional Edit* (By default this button is located just below the 3D view).

- Once you've turned proportional edit mode on, another button appears to its right, the falloff button. Select Smooth Falloff here. Alternatively you can use the menu on the header of the 3D View (*Mesh* → *Proportional Falloff* → *Smooth*) or, using Shift + O will cycle through all of the different falloff types while using the Proportional editing tool.

- Press G to grab the vertex. We should now have a circle surrounding the vertex, this is our *radius of influence*. Basically any vertices inside this circle will be affected by any changes to the vertex itself.

Noob Note: If you're having trouble seeing or changing the radius of influence, try saving your scene and restarting Blender.

- Use SCROLL or PgUp and PgDown to adjust the radius of influence to include just over 2 vertices on each side of our selected vertex. (Depending on your version of Blender, you may need to use LMB + SCROLL to adjust the radius of the influence. On Mac, use Fn + PgUp and Fn + PgDown).



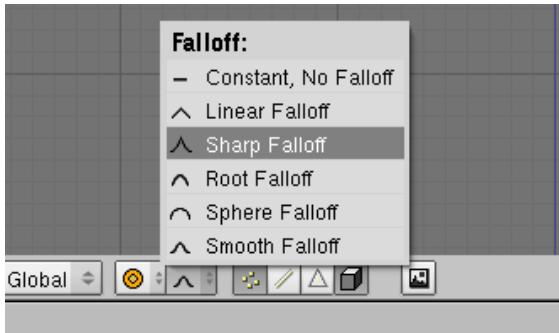
- Move the vertex up about 8 units on the Z-Axis. Do this by dragging the cursor up a little, and press the MMB ; this should restrain the movements along the Z-axis. Now use Ctrl to move it precisely. Alternatively you can use Z to restrain movements to the Z-Axis, type 8 and hit Enter . In older versions of Blender you may need to hit N before typing 8 .

Congratulations, we just created our first mountain. Now it's time to see what other things we can accomplish with the proportional editing tool.

2.26.3 Peaks vs. hills

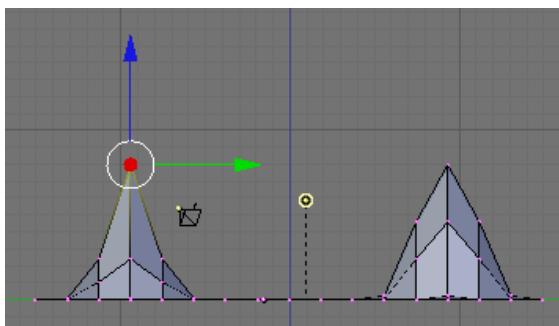
The 2.37 and onward releases offer at least 6 types and 2 modes of proportional editing. The previous release only has 2 of these types: Smooth and Sharp Falloff. We'll take a look at the difference between these two now.

- Change to top view again with Num7 . You'll notice that now your "mountain" looks like a few differently shaded squares in the grid; you're looking



down on shaded tiles, but in the Z axis, they're all still perfectly aligned with the original grid.

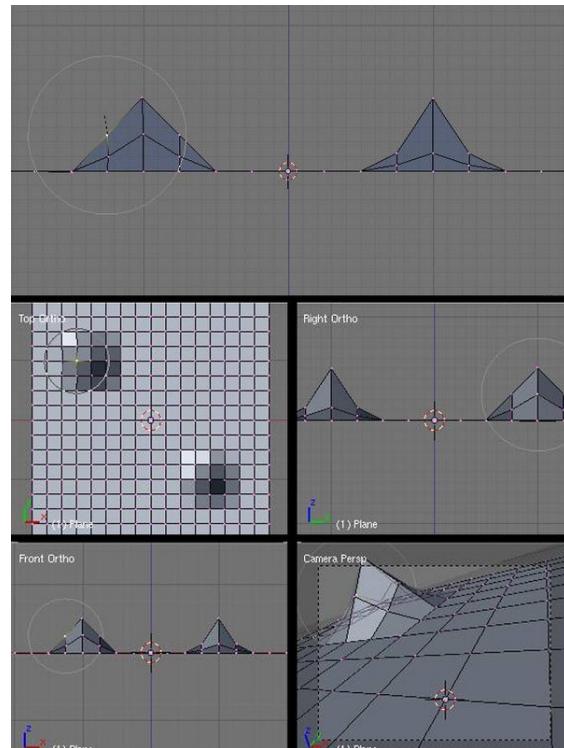
- Select another vertex away from the first. Let's say 4 from the bottom 4 from the right (counting the vertices on the edges).
- Change back to the side view with Num3
- Select Sharp Falloff from the menu on the bar of the 3D View. Alternatively, using Shift + O will switch from one to the next of the 6 proportional editing modes while using the Proportional editing tool.
- As before, move the vertex up 8 units on the Z-Axis (*Note: The radius of influence will still be the same size as when we last used it*).
 - G
 - Z
 - Type Num8 and hit Enter



Now we can see the differences between the sharp and smooth falloff. The same number of vertices are affected in both cases; only the degree to which they are affected is different.

The different proportional editing modes can be selected from the box immediately to the left of the proportional editing type box. The mode box contains four options: Disabled, Enabled, Connected, and Projected (2D). “Disabled” means that proportional editing will not be used. “Connected” means that only vertices linked to the selected vertices will be affected by the radius of influence. “Enabled” means that all vertices will be affected.

2.26.4 Shaping the world



Now that we've created a couple of Mountains, it's time to see how we can use proportional editing to shape them.

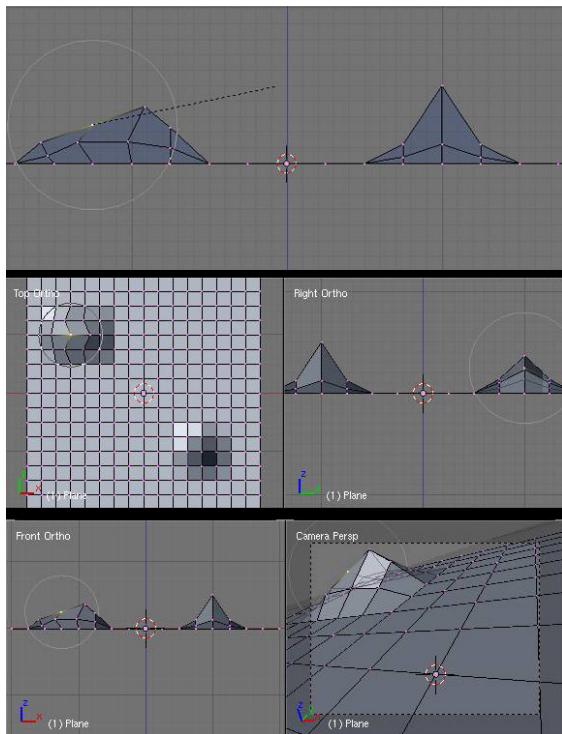
- First make sure we're in side view (Num3).
- Then on the smooth falloff mountain, the first one we created, select the vertex that is immediately down and left from the topmost point.
- Press R to rotate, scroll the MMB to change effective radius so it includes other points. Your screen should look like the photo to the right.

You can see the size of the proportional editing circle, and that there is only one vertex on the mountainside selected.

- Next hold Ctrl and rotate everything by -90. Alternatively, use R , N , and type **-90** and press Enter . Your mountain should now look like this:

Noob note: be careful about the range of affected vertices. If the range is too small, then rotating will affect just the selected vertex. If the range is too large, it will rotate everything together. You can adjust the range by using SCROLL .

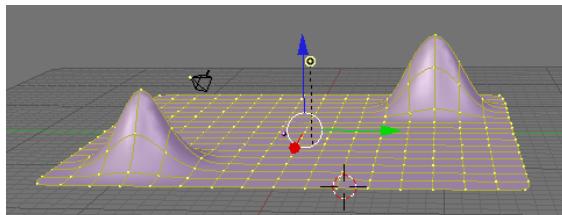
Notice that the vertex itself did not move; since it is at the center of the circle it had no effect. The adjoining vertices within the edit circle were rotated around it in decreasing amounts the further from the center they are. Try doing it again with a larger proportional editing circle. Feel free to play around with scaling or rotating from



different view points (don't forget that you can also use G to move vertices vertically or horizontally).

Try viewing your world from top view while rotating with a large effective radius. You will see the nearby vertices move close to the full amount while vertices further away move less.

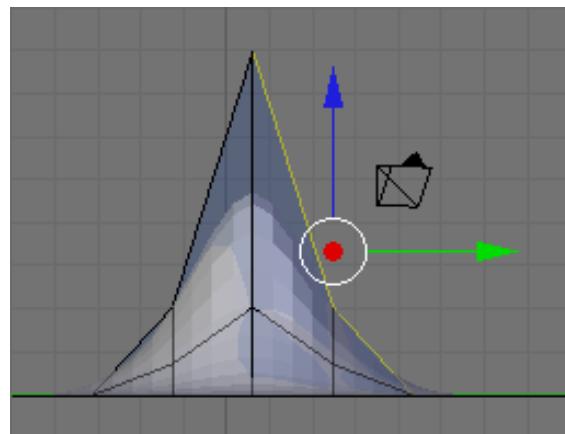
2.26.5 Smoothing things out



Now that we have a couple of budding mountains, you probably think they look kind of choppy. Sure they would be good if we were making an 8-bit console game, but we're working with 3D here, we want things to look sharper (or maybe smoother) than that. There are a couple of approaches to this. The first is to use more vertices when we create our plane. And I won't lie, it works. But it's also a HUGE resource hog. It would take your home computer hours of work just to keep things updated, let alone run it. So instead, we fake it. The easiest way to do this is to turn on *SubSurfaces* (we saw this in [Detailing Your Simple Person 1.](#)) For our purposes, let's set the subdivision (*Levels*) to 2. Also, ensure our SubSurf algorithm is set to *Catmull-Clark* (this is the default setting).

Now, you'll notice that with SubSurf on, we lose a lot of hard edges that we had, essentially we have no sharp corners any more. I don't know about you, but to me that doesn't make for a very interesting mountain range. So to restore our corners, we are going to use *Weighted Creases for Subsurfs*.

- First turn off proportional editing with O , and ensure we're in side view with Num3
- Next, while still in edit mode, change to *Edge Select* mode with Ctrl + Tab and select *Edges*. Alternatively press *Edge Select Mode* button at the bottom of the object window.
- In the Tool Shelf at left, select the Options tab, then under Edge Select Mode, choose Tag Crease.
- On our Sharp Falloff mountain, the second one we did, select the two edges on the right. (see image below)



- Press Shift + E or Space → Edit → Edges → Crease SubSurf, then move the mouse away from the edge until the edge *Crease* reads 1.000 in the 3D viewport header. If moving the cursor there seems to be impossible, just hit 1 and enter.

As you move the cursor away from the edge you will notice two things. The first is that the edge becomes thicker as we move from it; this is showing how much of a crease we have (with *Draw Creases* turned on). The second is that you will notice the subsurfed mesh moving closer to the edge as the sharpness increases.

2.26.6 Naturalness

Press Ctrl + Tab to enter Edit Mode and select vertices. Then go into front view Num1 . Select the second vertex from the top in the centre of our Sharp Falloff mountain, then go into side view Num3 . Hold G and drag the vertex inwards, not too far or your mountain will come out of

itself on the other side. Just bring it in enough to make a small indent.

Then grab the top vertex and pull it down a small amount. You will notice that there is a small “crunch” in your mountain.

Don't forget to select all with A , then W Shade Smooth button to smooth everything out.

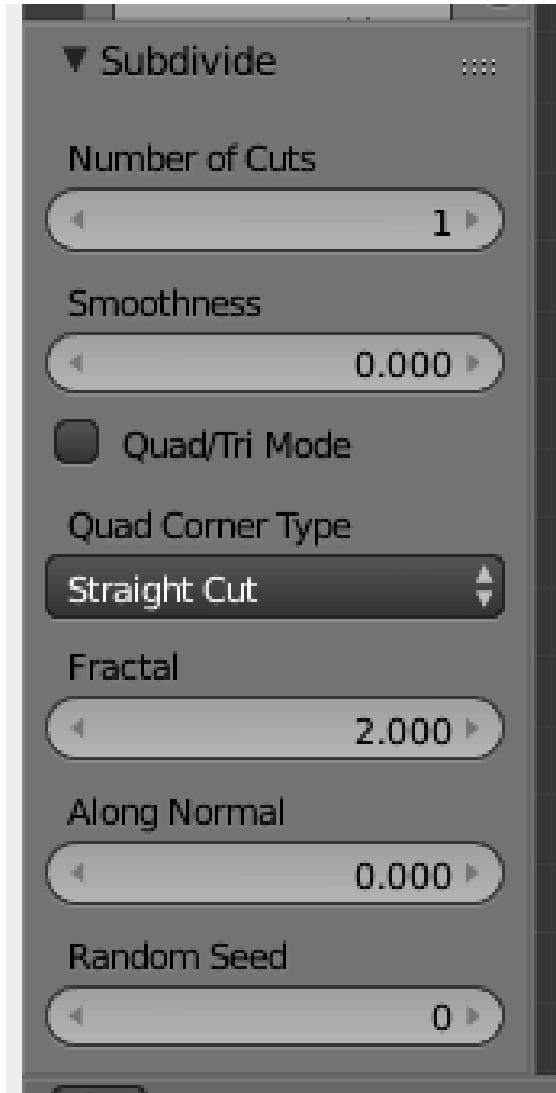
OK, so your mountains are starting to shape up. But they still look a bit too neat. You could spend time moving each individual vertex but the chances are your model will still lack the natural feel. What we need is some chaos. Thankfully this is quite easy to accomplish. Firstly select the vertices that make up your mountains, all of them and a few around the base (box and circle select will make this easier). Select a few vertices between the mountains too. Next we use something called fractals. Fractals are chaotically (i.e. randomly) generated variables. In short you can use these variables to give your mountains a “wobbly” look.

In the Tools tab of the Tool Shelf, press Subdivide (under Mesh Tools), then look at the Subdivide submenu below. The value in the Fractal box is the strength of the fractal. 1 is very low and will barely change your model. 10 is very high and will twist your models into very odd shapes indeed. Have a play with different values until you find one that you like. Around about 4.0 should do it. Hit OK and presto, your mountains have been transformed from clinical neatness, to lumpy chaos.

- If you make too many fractals, your computer will slow down. However, the more you add, the more bumpy and realistic it looks!

Repeatedly using the fractal tool seems to rapidly multiply the amount of vertices on your canvas. I suggest using the tool once, and if the result isn't satisfying, undo the result (Ctrl + Z) and try it again with a different fractal strength. Helpfully, even after undo, your selected vertices remain selected.

Now go back into Object mode and view the result.



Fractal option in 2.72

2.27 Modeling a Volcano

In this module, you will create a volcano using the proportional edit fall-off tool. You should be comfortable with deleting and adding meshes.

2.27.1 Adding a Plane

Delete the basic cube. Add a plane, and S scale it up by 10. Rotate it so you see it in top-view (make sure it's in Orthographic view too).

Enter Edit mode and subdivide (with W) 5 or 6 times. More subdividing will give you a “smoother” volcano, but it also needs more CPU power.

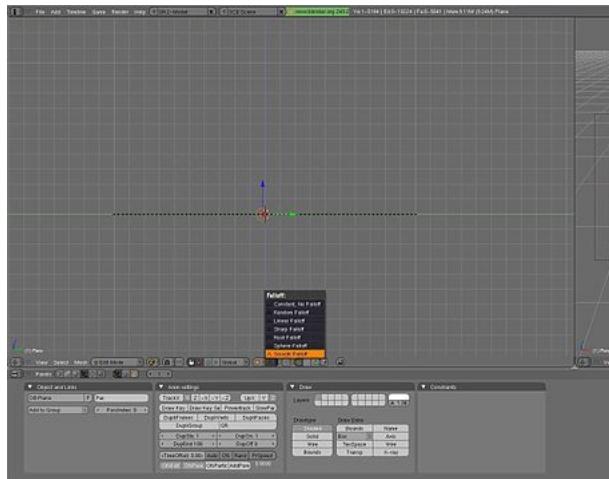
The difference between “Subdivide” and “Subdivide Multi”...

“Subdivide” divides every square in the plane into four new squares. So every time you press “Subdivide” you will have four times as many squares as before. “Subdivide Multi” will make x horizontal and x vertical lines through your existing squares, so the new number of squares is: $(\text{squares}_{\text{old}})^*(x+1)^2$, where x is the number you enter.

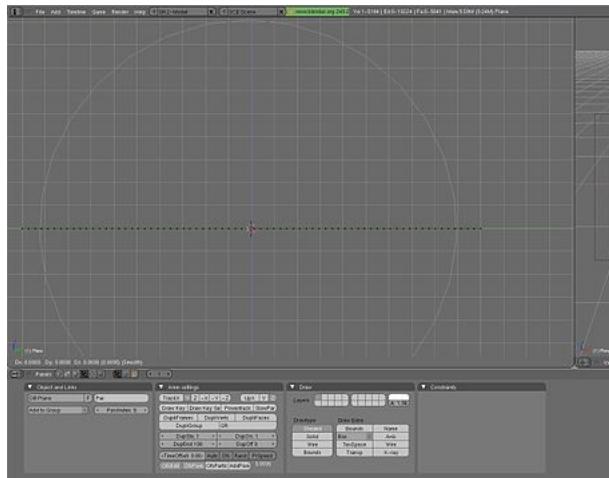
2.27.2 Making the Mountain

In top view, select one of the points in the middle of the plane. With this point selected change to side view. Press the O , which enables the “Proportional Edit Falloff” tool in the Menu-Panel beneath the 3-D-Window. As seen in

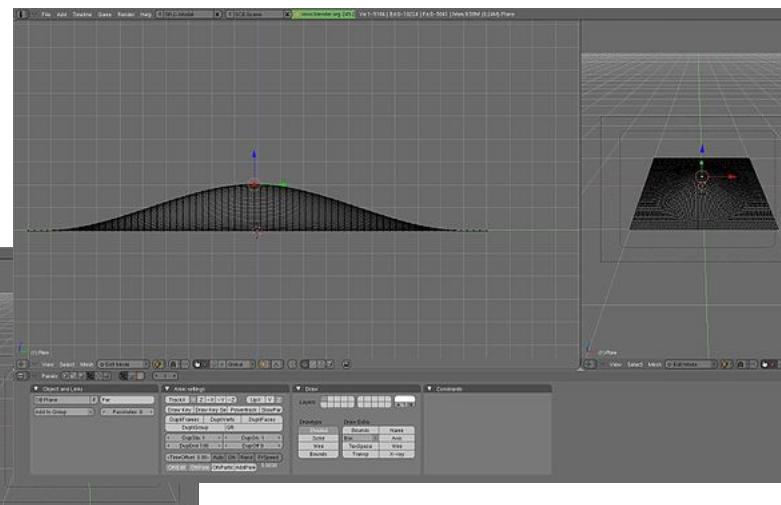
the previous tutorial **Blender 3D: Noob to Pro/Mountains Out Of Molehills** when you move a vertex while edit falloff is enabled, all vertices in a defined radius of the selected vertex will align with the selected vertex when its position is altered. How they are adjusted can be chosen in the tab on the right of the yellow dot. I propose using “smooth falloff”.



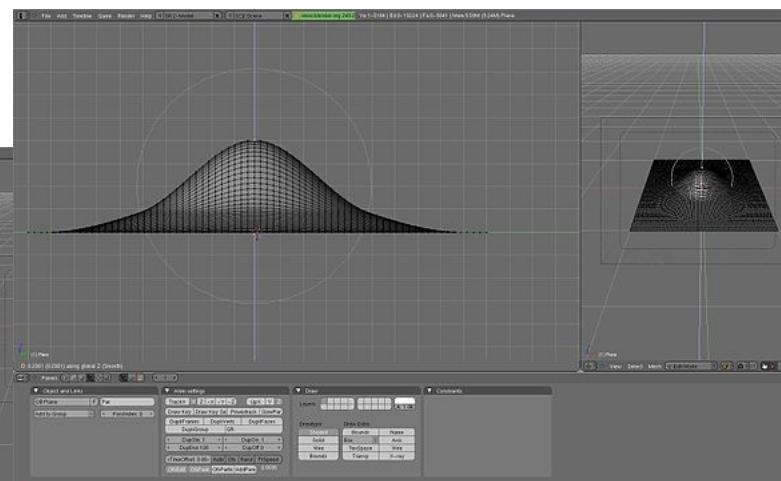
Now grab the vertex with G . You will now see a gray circle. You can change its size with the mouse wheel. Every vertex inside this radius will be affected by the falloff. Change the size of the circle so almost the whole plane is in it.



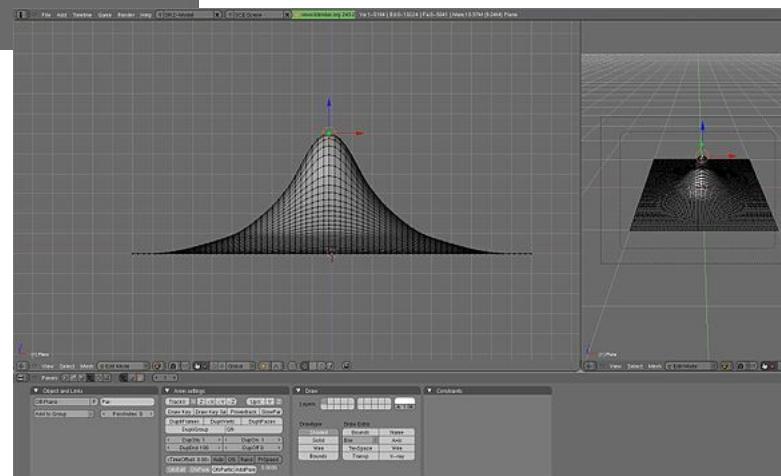
Now move the vertex a bit upwards, as seen in the picture. Optionally you can lock the z-axis to make the volcano go straight up by pressing Z .



As you can see all the other vertices will shift upward. We could keep moving this vertex at the same rate, but that would cause the plane itself to rise and bend, and that's not very good. So press LMB to apply the changes, grab the same vertex a second time and repeat the previous exercise as before, except now choose a smaller radius for the circle, about half the diameter of the plane (G → Z → scroll MMB).

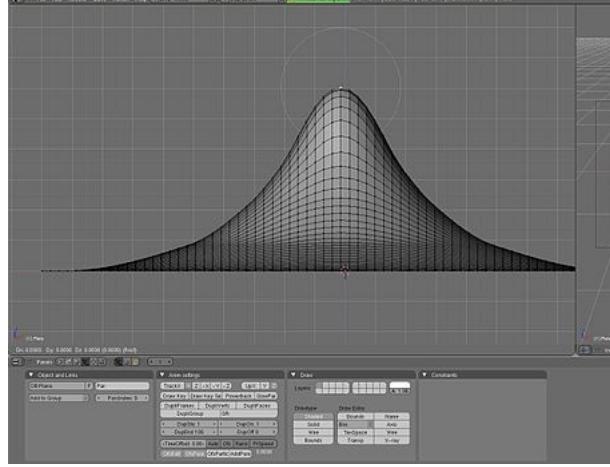


Repeat this two or three more times and you will get something like this:

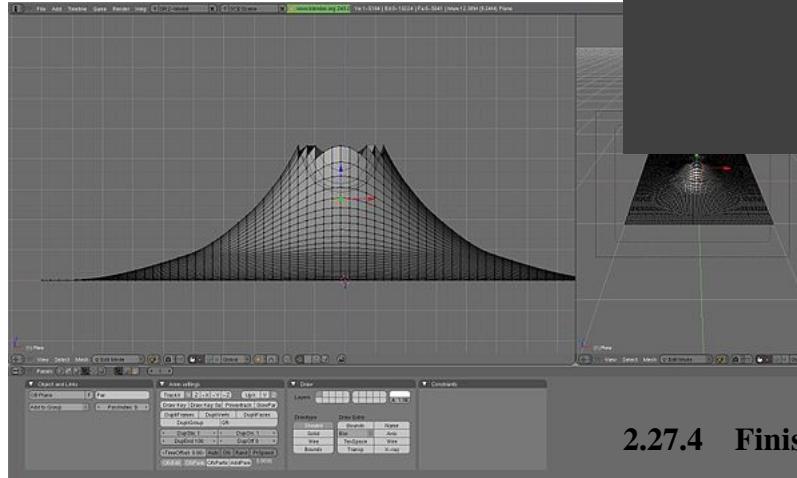


2.27.3 Forming the Crater

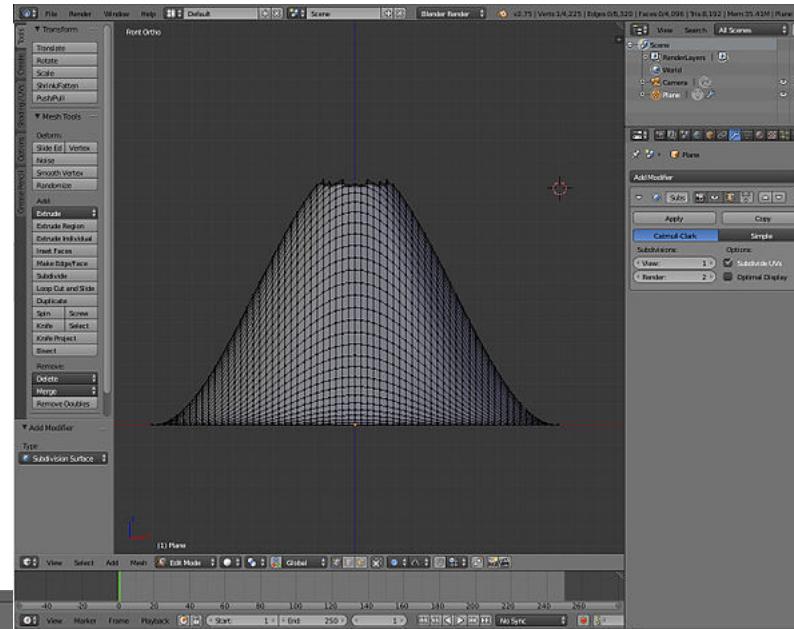
Now we're going to create the "hole" on the volcano. First change the falloff to "root". Grab the vertex one more time, change the size of the circle so it's more or less as seen in the picture.



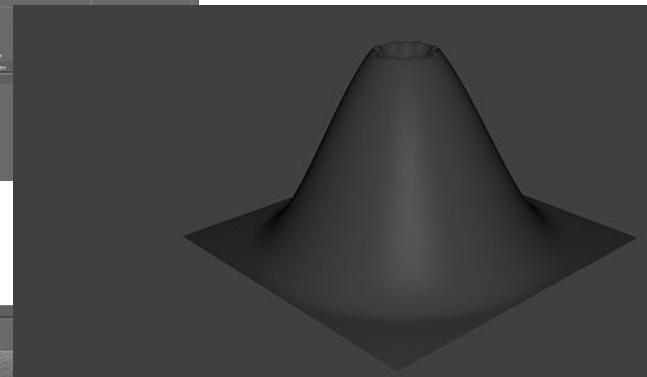
Grab this vertex down a bit, apply, grab it one more time with a smaller circle. You now should have something like this:



Just leave the border jagged and just smooth (Subdivision Surface) the whole volcano cause it is much more realistic. Go to Object mode, select the volcano, go to the "Modifier" menu in the "Properties" Header and just click on "Add Modifier" -> Subdivision surface (you can leave "view" on 1). Do not apply these settings yet.

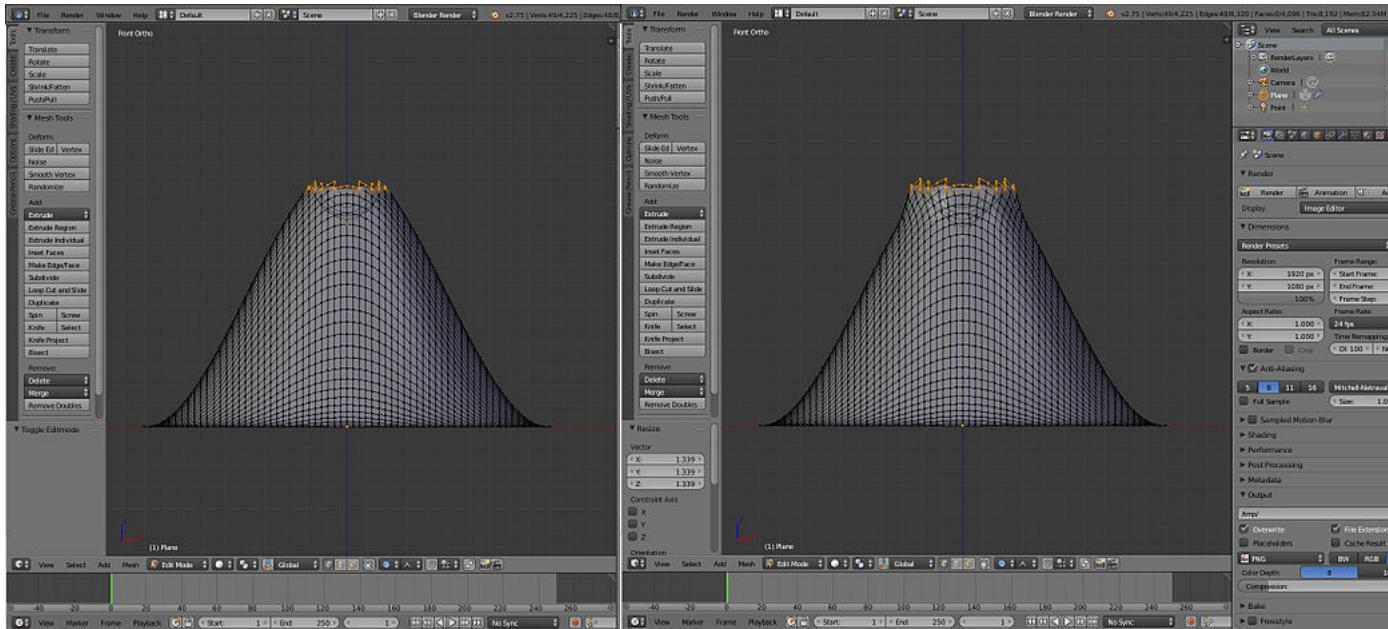


First we'll do a test-render. Still in "Object mode" Delete the default Lamp point with "X" or "Delete" and place your 3D cursor behind the camera and press Shift + A -> Lamp -> "Point". With the Lamp Point still selected Click on The Lamp point Properties ("Data") in the "Properties" Header then change "Energy" to "10". Press F12 to enter Render, after adjusting the camera.

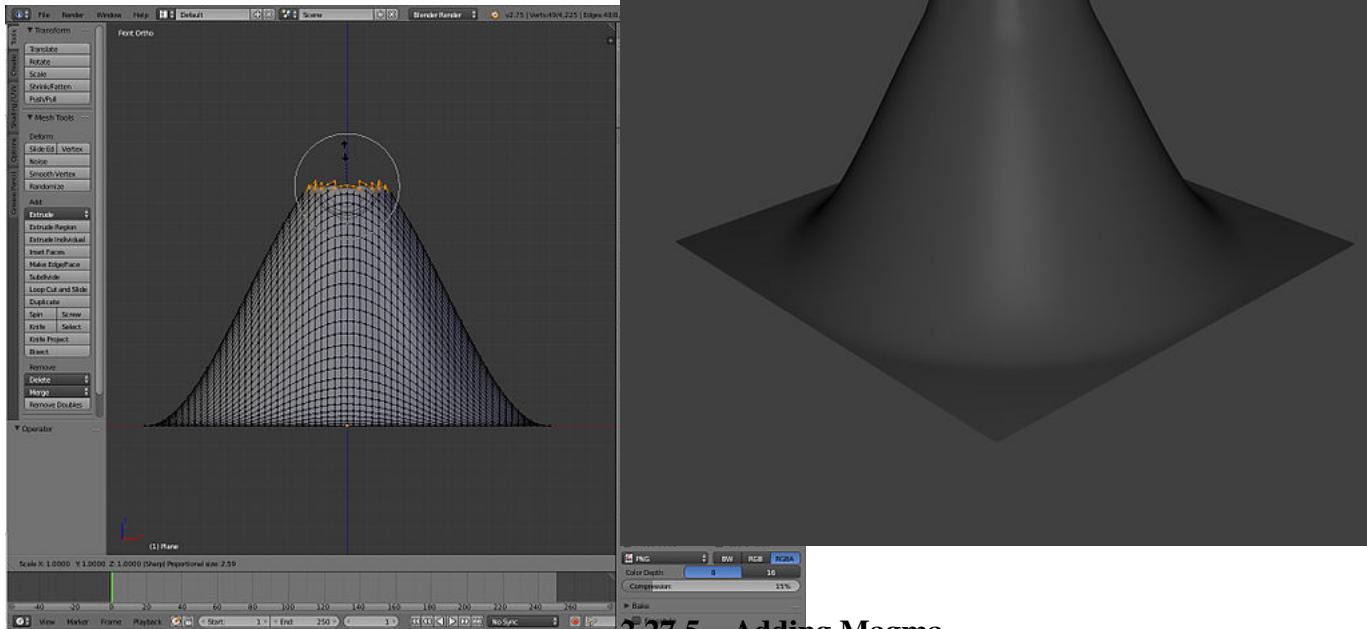


2.27.4 Finishing the crater

You can very easily make a nice looking crater. Just go into "Edit Mode", touch "Num1". Make sure "limit selection to visible" is off and "proportional editing" is on and set it to "sharp" falloff. Select about the upper vertices with "border select" (Press "B key").



After that, scale (press “S key”) it ‘till it’s a nice crater with a circle as large as mine.



2.27.5 Adding Magma

And that’s it, you just created a nicer looking crater.

Let’s add some “magma” using lighting.

1. Make sure you’re in “Object Mode”
2. Press Shift + S and choose *Cursor to Center*.
3. Press Shift + A and choose *Lamp → Point*.
4. In the Properties window, click the Data tab.
5. In the colour box (white by default) in the Lamp section, change the color to reddish-orange. (Red: 1, Green: 0.1, Blue: 0)
6. Set the Energy to around 7.

7. Raise the light until it's just above the bottom of the crater (G rab along the Z axis).
 8. If the ground level of your plane is reflecting light from the lava lamp this is because the bottom of your crater is above ground level of the plane you created; you'll need to turn on ray-tracing. in the object data menu for the light, open the Shadow menu and click "Ray Shadow"
- Alternate 1: Spot Lamp

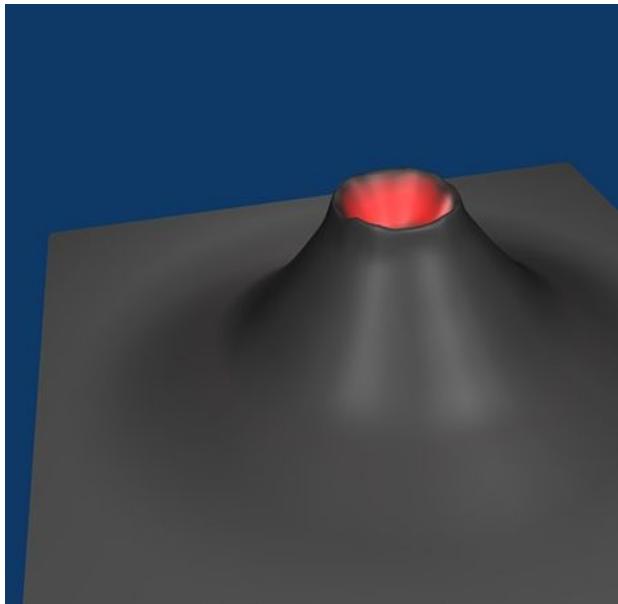
1. Change the light's type to *Spot*.
2. Raise the light until it's covering most of the crater. If the light is not pointing down, R otate and angle it downwards. You can also scale the radius of the light by press S to fit the rim of the crater.

- Alternate 2: Area Lamp

1. Change the light's type to *Area*.
2. R otate along the Y axis: 180 degrees.
3. Set Gamma to 2.
4. Set Distance to around 5.

Experiment with the values and positioning to get something that works with your volcano.

It should now look like this:



2. Select the "Material" button  and press New.
3. Change the Diffuse color to ashen gray. (Red: 0.260, Green: 0.230, Blue: 0.230)
4. Select the "Texture" button and press New.
5. Change the Type to Stucci.
6. In the Influence panel, uncheck Color, and check Normal. Set the Normal slider to 0.5. This will render the texture as a bump-map.

(Note: In version 2.77 you may need to change the texture Mapping -> Coordinates option from UV to Generated before you see bumps appear.)

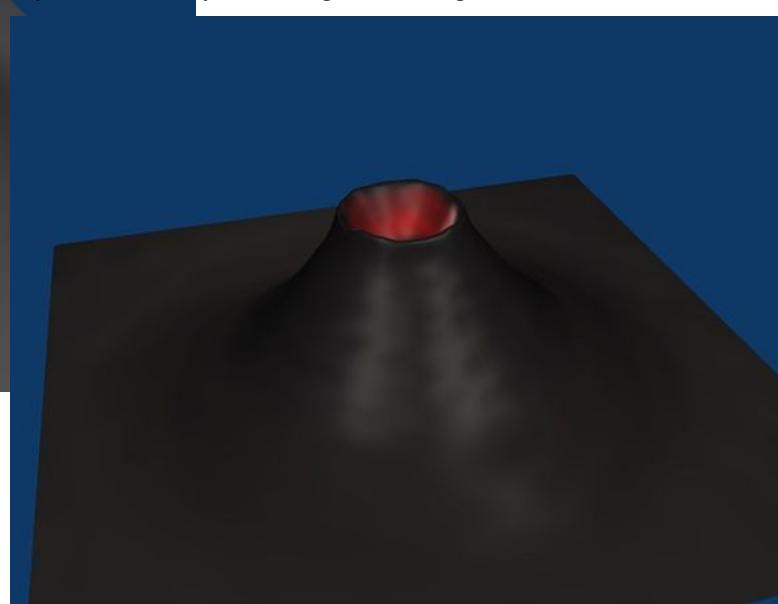
(Note: In version 2.78c you may need to change the texture Mapping -> Coordinates option from UV to Global or Object before you see bumps appear.)

Older versions:

Select the volcano and press F5. Keep Pressing F5 until the Materials Buttons (symbolized by a red ball) is highlighted. Then add a new material. You do this by clicking the Add New button in the Links and Pipeline Panel. Once you've done that, set the settings similar to the picture below. Now press F6, then add a new texture to the material. Choose a stucci texture, set the noise size to 0.15. Now switch back to the materials-window (F5) and click on the "map to" tab. Deselect the "col" button and select the "nor" button. This will render the texture as a bump-map on the volcano. Set the "nor slider" to 0.5, which should be the default. Switch to the "map input" tab and choose "tube".



If you now render you should get something like this:



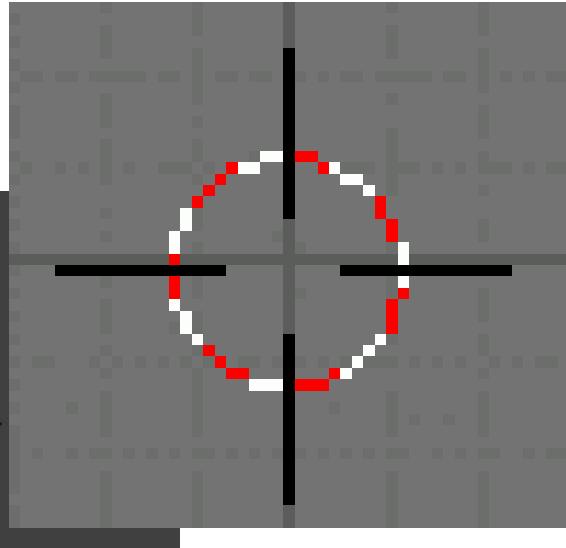
2.27.6 Varying the Terrain

Next, let's set the volcano's material.

1. RMB on the volcano plane.

This looks really smooth, like clay pottery. To get a more rough-looking volcano, try out these options:

- Option 1: Subdivide and increase the fractal. 5 should do the trick.



- Option 2: Decrease the texture's basis size (in the “Stucci” panel when you select the “Texture” button).
- Option 3: Proportional edit tool.
 1. Press O to turn on Proportional Editing mode. Select Random fallout.
 2. Grab the center vertex, and raise it along the Z axis.

Older versions:

Go into Edit mode, select all Vertices, and use the fractal (set from 15 to 30) to really get things looking rocky and mountainous. TAB (Edit mode) → F9 → AKEY to select all → Mesh Tools → Fractal → 15 - 30 (15-low, 30-high) → OK → TAB (Object mode) In Blender 2.5 you can use the random proportional edit tool: use NUM7 to switch into top view, grab the central vertex of your volcano using a large-radius random proportional edit, and pull it slightly into Z-direction.

(A note: Seems there is no need in subsurf at all since fractal tool will dramatically increase vertex quantity.)

2.28 Penguins from Spheres

Note: Some Pictures are outdated.

2.28.1 Setup

Start with the **default scene**: it should contain a selected cube. Delete this cube by pressing X → Delete.

Put the 3D cursor at the scene center by pressing Shift + C .

Note: after deleting the cube you must be in Object Mode. If not, Press CTRL-Z and switch with TAB and redo the operation.

2.28.2 Creating the body

Noob note: to ensure that you don't become confused, make sure that your viewport is set up in the same direction you see in these pictures. The colored arrows are red, green, and blue and they control the x, y, and z axes, respectively.)

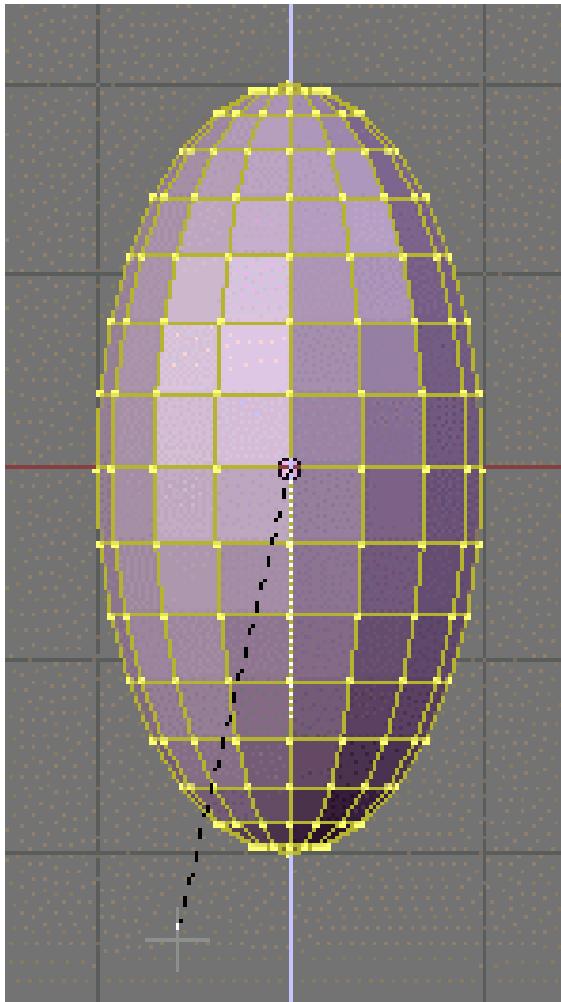
- We start by creating our main body from a sphere. Press SHIFT + A → Mesh → UVSphere, then choose 16 segments and 16 rings.

We're going to make it look like a penguin body:

- Press TAB to enter Edit mode
- press NUM1 to switch to the front view,
- with all the vertices selected (if not, A), choose the scale tool (S),
- restrict scaling to the Z-axis (Z),
- and move the mouse away from the 3D cursor while holding down the CTRL key (this snaps the scale values to whole numbers),

• **Note:** Make sure the mouse cursor is not too far away from the sphere when hitting the S or else you may not be able to reach a 2.000 scale value. The scaling steps are proportional to the distance from the 3D cursor when calling the scale tool.

- the current scale shows in the lower left corner of the viewport, click when you've reached 2.000 LMB).



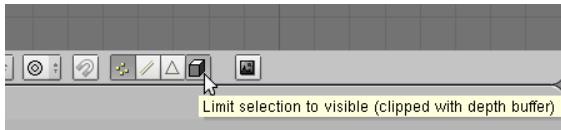
Z-scaled sphere

- **Note:** You can also do this by typing in 2 after starting the scale and restricting movement to the Z-axis: S -> Z -> 2

This is our main body!

2.28.3 Shaping the head

We're going to shape the penguin head from the top of the sphere.



Start by selecting the top-most single vertex as well as the top two smallest circle segments.

Note: Selection has been explained in a previous tutorial. Here, the easiest methods are either box selection B) in the front view NUM1) and

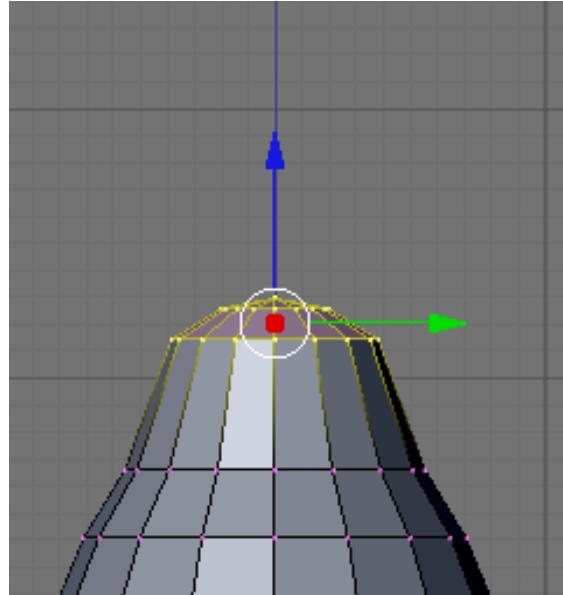
Limit selection to visible off, or lasso selection (CTRL + LMB) in the top view (NUM7) and Limit selection to visible on. You can also switch to top view, center your mouse on the top-most vertex and use the circle tool. Don't forget to deselect all first (A)

Noob note: You can also select the top vertex and press CTRL + NUM+ twice to select the circle segments.

Noob note: Make sure the *Limit selection to visible/Occlude background geometry* button is in the right state each time you select vertices, edges or faces. When it's off, selection affects any item, visible or not.

Building the neck with the 3D transform manipulators

To turn on the 3D transform manipulator, either push down its button or use CTRL + SPACE and choose Enable/Disable.

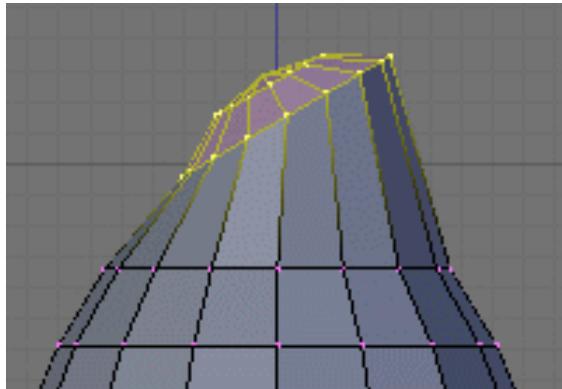


Moving the two selected circles up Go to the front view (NUM1). Drag the blue arrow while holding the CTRL key down to move the selected vertices 0.3 units up.

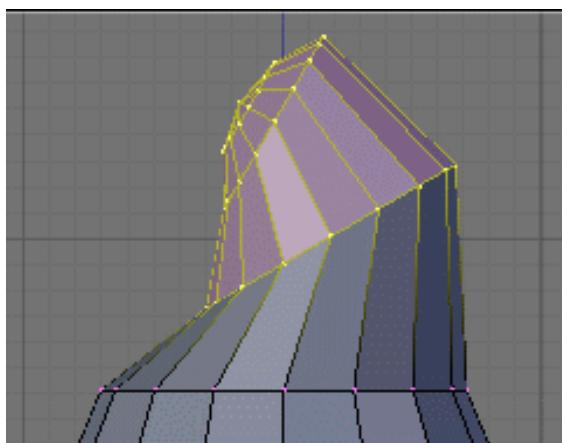
Noob note: you may not be able to snap the extrusion lengths to tenths of units. CTRL snaps to the grid size by default: if you can only translate by one unit (1.0), zoom in until the grid divides itself into tenths (**SCROLL**). Some Blender versions allow to snap to one tenth of

the current step by holding both the SHIFT and CTRL keys while moving the mouse.

Noob note: instead of the *Transform* manipulator, you can use the G and constraint the movement to the Z axis (Z).



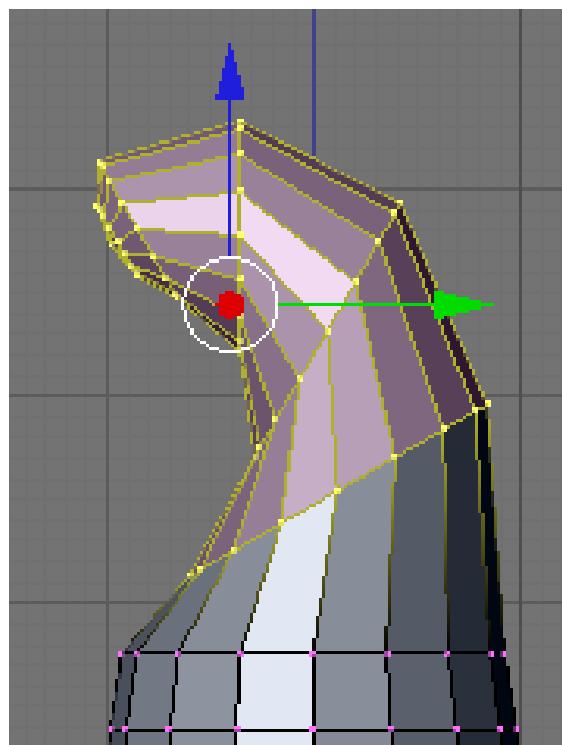
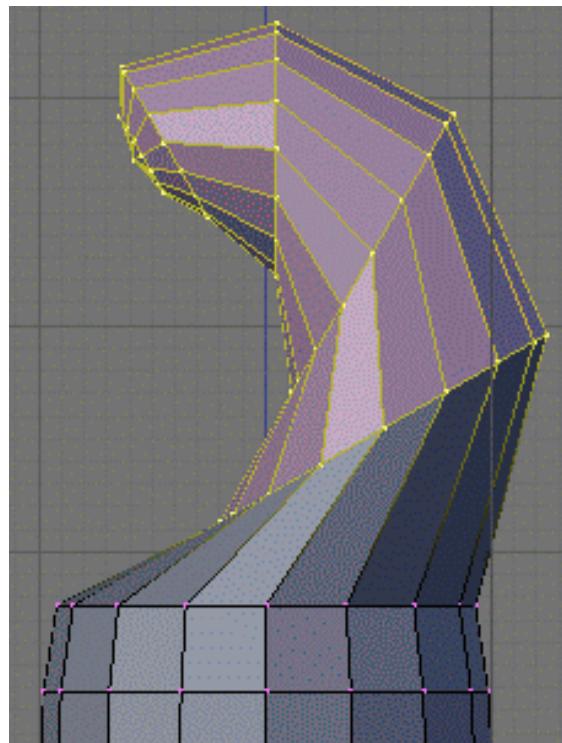
Rotating the neck Now switch to the side view (NUM3) and make sure that the rotation/pivot point is set to “median point” either by selecting it from the third drop down menu right of the “Mesh” menu, or by pressing CTRL + . Choose the *Rotate* tool (R). Move the mouse with the CTRL key down to rotate the selection 30 degrees counter-clockwise. Use LMB to validate the rotation.



Select an additional ring of vertices by expanding the selection (CTRL + NUM+ (Note: NUM+ Refers to the addition symbol on the NUM Pad, KEY+ will not do.). You can contract the selection by pressing CTRL + NUM- . Move these vertices an additional 0.3 units up, then rotate them as previously 30 degrees counter-clockwise in the side view.

Repeat those steps (selection expansion, translation and rotation) two more times and you'll end up with the body seen in the below.

That doesn't really look like a penguin, yet!

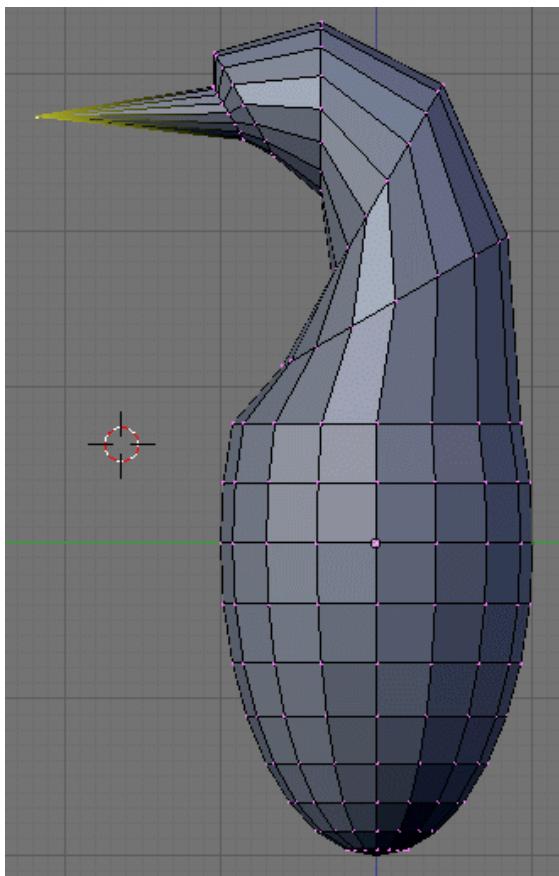


Now move all of the selected vertices to the left 0.4 units by pulling the manipulator's green arrow (and of course holding the CTRL key down). This straightens out the neck as seen in the next picture.

Noob note: you can also use the G and translate the selection by -0.4 as displayed in the bottom left corner of the viewport. Still do this

in the side view (NUM3).

Noob note if you had to pull the red arrow, not the green one, then you probably didn't switch to side view, and modeled the neck in front view. If you realize here that you have this done from another view then press AKEY twice, then NUM7, then Space -> Transform -> Rotate and type 90 or 270



Creating the beak Switch to the front view (NUM1), and select the frontmost vertex (the one that originally was the top vertex of the sphere) with the RMB . Then switch to the side view (NUM3) and translate this vertex to the left by 1.2 units using the manipulator's green arrow or the translate tool.

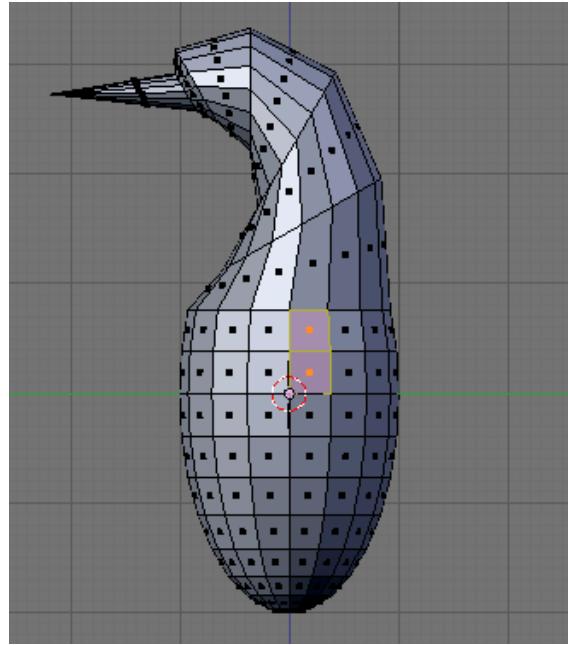
Note: some Blender versions allow moving the vertices from the keyboard with the following sequence: G , Y , -1.2, ENTER .

The main body of the penguin is now finished. The next step is to create some flappers for the poor little guy.

2.28.4 Extruding the wings

We are going to create the wings by extruding faces on each side of the penguin.

Noob Note: You can rotate the whole object by pressing A to select all of its vertex and then rotate it pressing R or using the Rotate Manipulator until the axis on your screen matches the axis on the example image. That way, also, you can practice a little more



Choose the side view (NUM3) and switch to the *Face select mode* (CTRL + TAB → Faces, or click on the orange-sided cube icon in the toolbar).

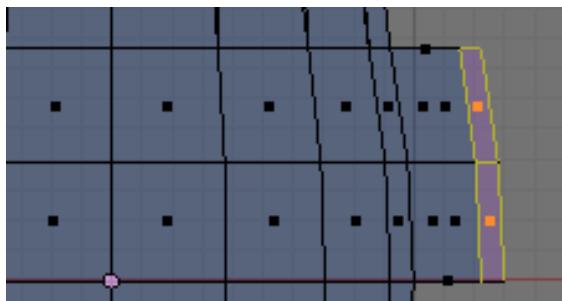
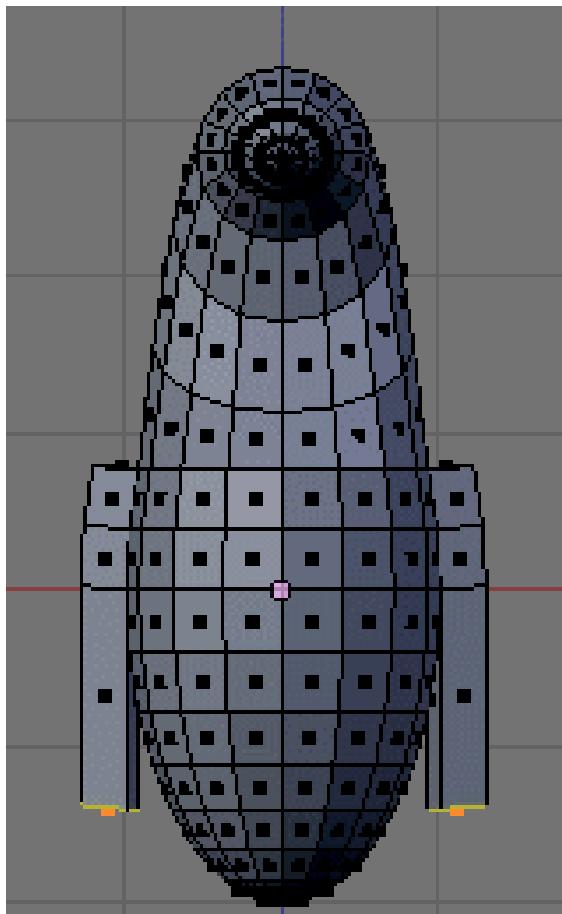
Now, select two faces that will make up the penguin's shoulder as shown on the right (either select the first one with RMB and the second one holding SHIFT , or use box selection (B) to select them both in a single operation).

Then switch to front view (NUM1) and extrude the selection:

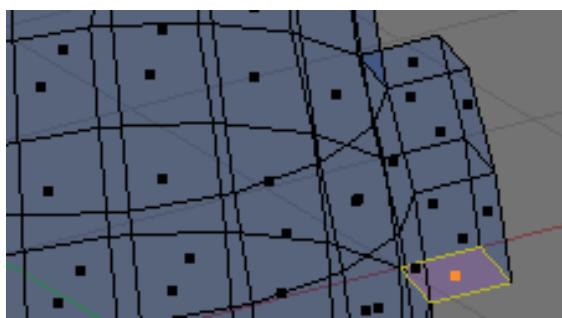
- choose E → Region,
- constrain to the X axis (X),
- hold CTRL to snap,
- and move the mouse to extrude the shoulder by 0.2 to 0.3 units.

We'll now extrude the bottom face of this new extrusion. Rotate the view to show it with:

- a MMB drag,



Wing extrusion



Bottom face selection

- or several presses on NUM2 ,
- or CTRL + NUM7 (bottom view).

Press the A to deselect all, and select the bottom face (RMB), switch to front view (NUM1), extrude by 1.4 units down (E and CTRL).

Now do the same on the penguin's other side: use CTRL + NUM3 to view the left side (you can also rotate with MMB , or press NUM4 several times).

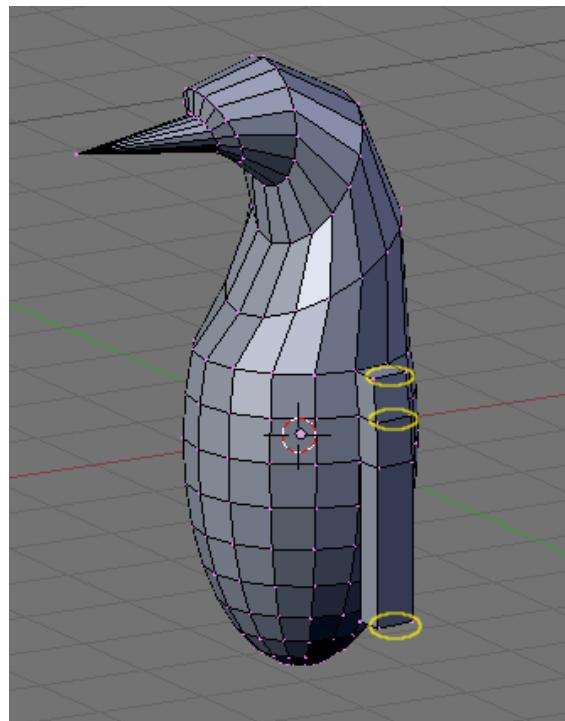
2.28.5 Smoothing the wings

We're going to smooth out the shoulders and improve the wings. Though this can be done in many ways, we'll only use the merge tool.



Rotate your penguin so that you can see one shoulder from above. Then switch to *Vertex select mode* (CTRL + TAB → *Vertices*). Press A to deselect all, then select the two shoulder vertices with RMB and SHIFT .

Press ALT + M , choose *At Center* from the popup in order to merge the two vertices at their center. Finally dismiss the message saying *Removed 1 Vertices*.



Edges to smooth

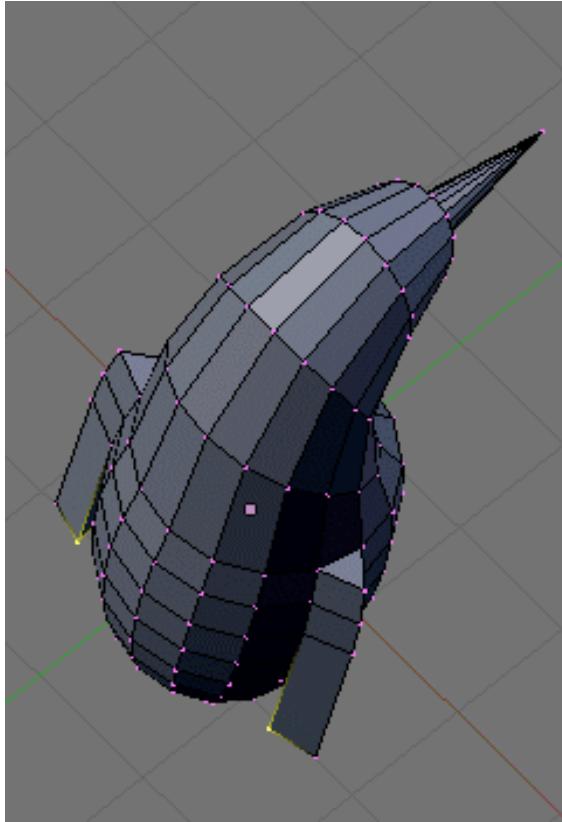
Repeat the steps with the two other vertex pairs shown on

the right picture, and smooth the other wing. I'm leaving the middle segments for now, else the wing tips will be too pointy.

Note: if you have troubles merging vertices, it comes from vertex duplicates in your mesh. You probably chose Individual Faces instead of Region when extruding the wings, which creates duplicate vertices and neighbouring faces. To clean up your model: select all vertices (A) and choose W → Remove Doubles.

You must do them one by one!

Or, Alternatively, change into Edge select mode (CTRL + TAB → Edge), select the edges to smooth (CTRL + LMB), press ALT + M , choose Collapse

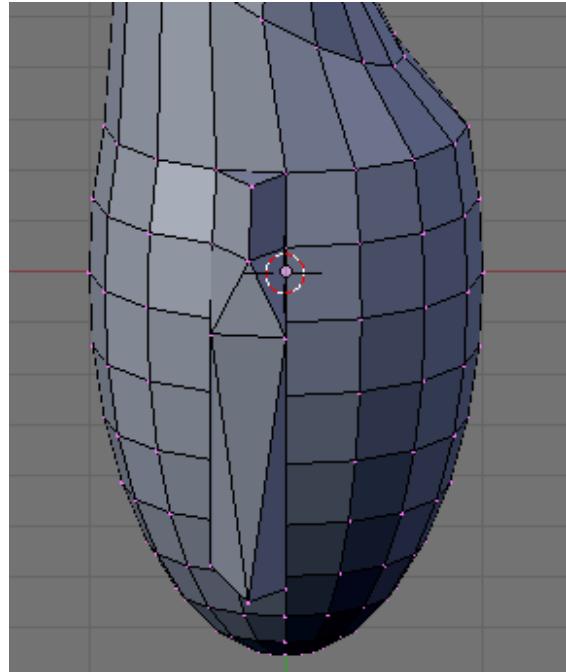


Finish off the wings by selecting the two backmost vertices of the wings, and moving them up using the blue arrow by 0.1 unit.

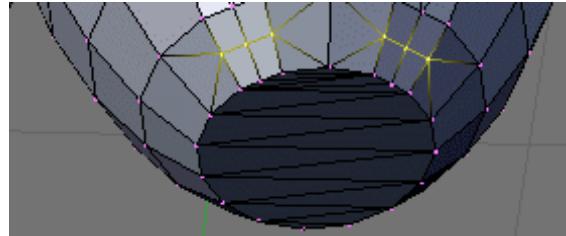
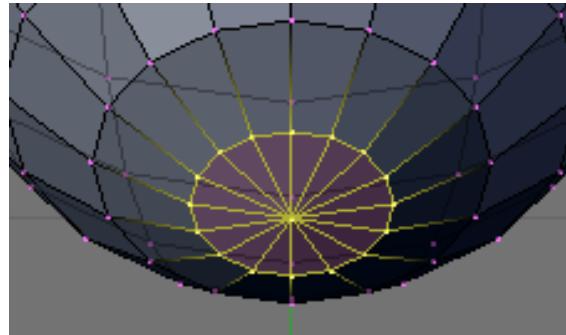
You should now have something like this:

2.28.6 Cutting the underside

We're going to cut the penguin's lower end, for it to stand up! Select the bottom vertices (bottom vertex and the first ring above it) as shown in the picture. There are many ways, this is left as an exercise.



Wings complete

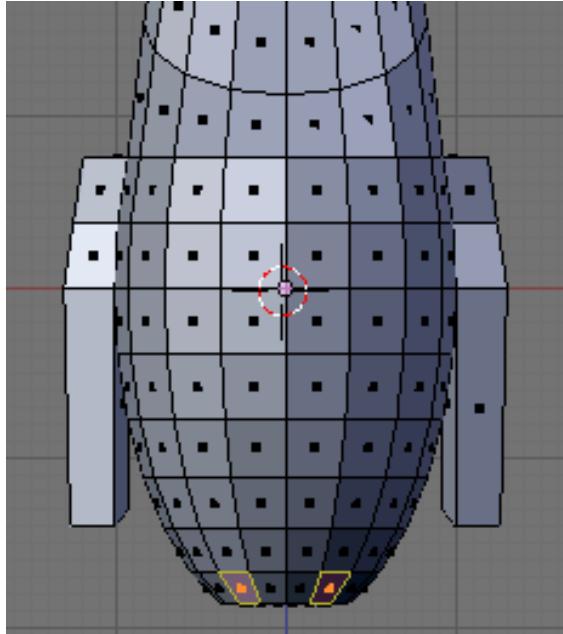


Once they're selected, delete them (X → Vertices). Now our penguin is hollow: select all the vertices around the hole, and fill it using ALT + F .

Noob note: To quickly select all the vertices around the hole, you can enter edge mode CTRL + TAB -> “Edges”, and then select one edge that goes around the hole. Now press **CTRL+E** -> “Edge Loop Select”, which should select all edges around the hole. Now go back to Vertex select mode and continue with ALT + F .

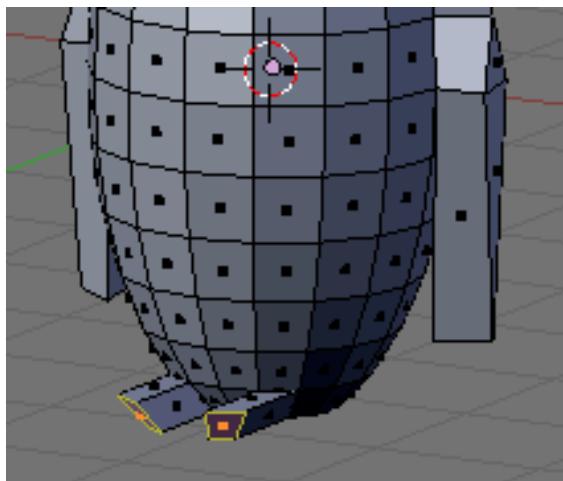
Noob note: For Blender 2.56 In vertex select mode and occlude on hold down ALT and select one of the vertices all the vertices in the ring will be selected.

2.28.7 Adding the feet

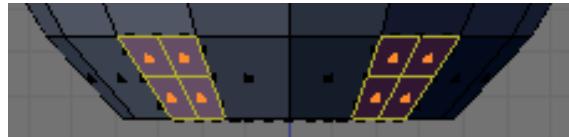


The next step is to provide the little guy with feet. To do this, we're going to extrude two of the front faces:

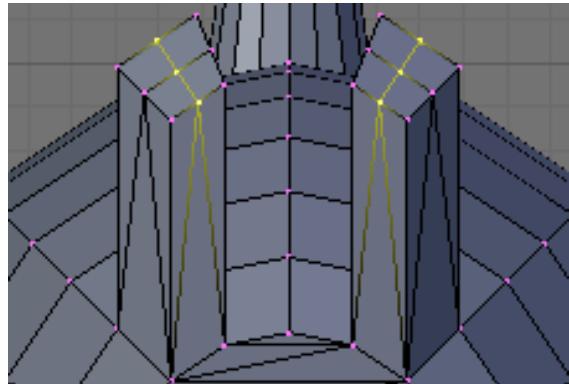
- choose the front view (NUM1),
- switch to *Face select mode*,
- turn on *Limit selection to visible*,
- and select the face to the left and right of the middle two faces of the penguin.



Then switch to the side view (NUM3) and extrude the selection by -0.6 units (E → Region, restrict to the Y axis: Y).

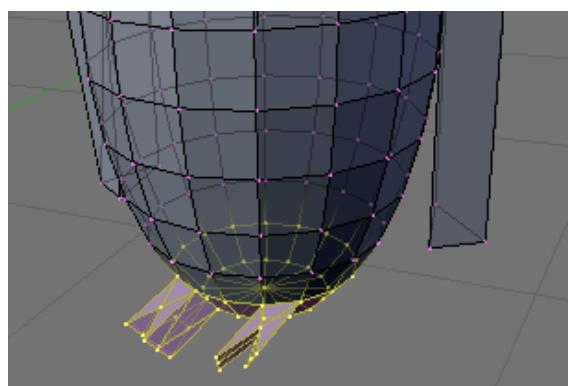


Keep the selection and look for the *Mesh Tools* in the Tool Shelf. If you can't see it, press T to make the Tool Shelf visible. Then click on the *Subdivide* button (under Add in the Tools tab). Or press W and choose Subdivide.



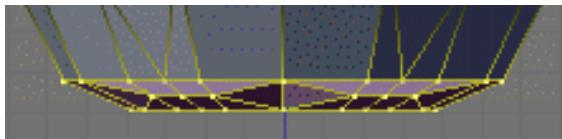
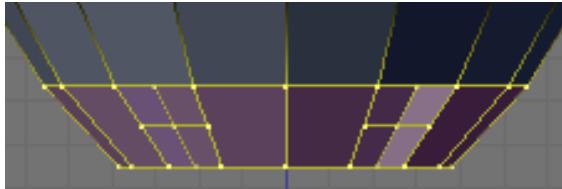
Switch to “Vertex Select Mode”, Now select the three middle vertices (or two edges) vertically at the tip of each foot, and drag them along the Y axis by 0.3 units towards the penguin

Note: if something goes wrong here, you may need to remove double first. As always, to move the vertices, either use the manipulator or G and Y sequence.



You should end up with what's shown in the right picture (minus the selection).

The feet look too thick, let's flatten them a bit. Switch to the front view (NUM1) and select the two bottom vertex



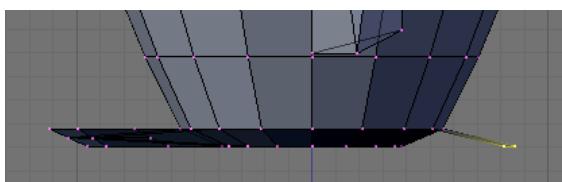
rows (*Limit selection to visible* off, use either the lasso or box selection).

Then choose the scale tool (S), limit its action to the Z axis (Z) and scale down by a factor of 0.4.

The feet still look rather peculiar, so please go ahead and move the vertices around on your own as you like.

Reminder: you can use the G and restrict movements to the X or Y axis using the X or Y. Try not to move vertices along the Z axis to keep the penguin's bottom flat.

2.28.8 Extruding a tail



To complete the penguin, we have to add a tail (the end of the tuxedo):

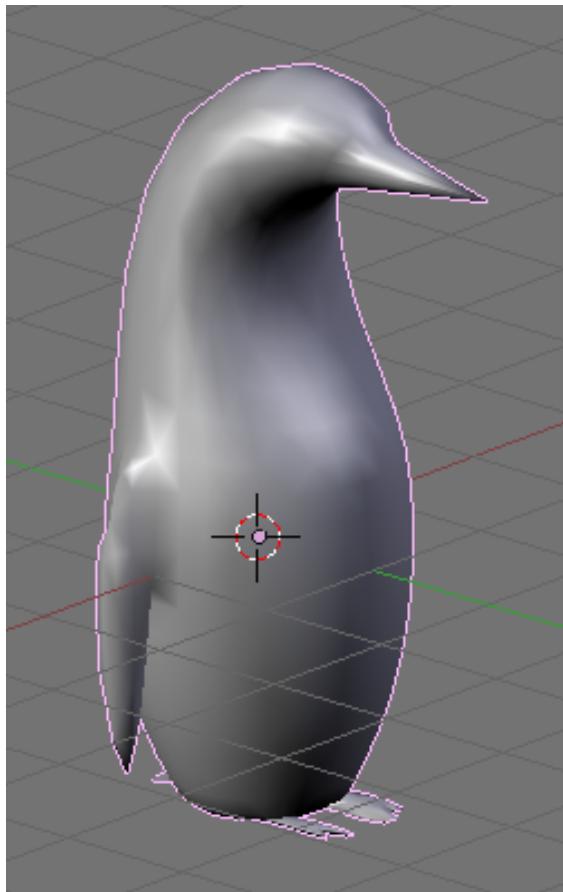
- go to the back view (CTRL + NUM1),
- make sure you're still in *Vertex select mode*,
- and that *Limit selection to visible/Occlude background geometry* is on.

Select the three middle vertices in the second row up from the bottom. Then, switch to the side view (NUM3) and extrude the edges 0.3 units away from the penguin and 0.08 units down (E → Edges), so that the end of the tail is at the same level as the bottom of the penguin.

Noob Note: Or you can extrude the edges 0.3 units away from the penguin, then G Z –0.08, LMB or ENTER .

Press CTRL + S to save your work!

2.28.9 Subsurfing



Now that's what I call a penguin. But then again, there are no penguins where I live, so I might be wrong.

Go to *Object Mode* (TAB), and make sure the penguin is selected. Then check for the *Modifiers* toolkit in the Buttons panel. Press *Add Modifier* → *Subsurf* (or Press SHIFT + O).

Look at the penguin now, he's much smoother. You can alter the levels of the subsurfing if you like, but I'll settle for level one. Under the *Links and Materials* toolkit, you can press the *Set Smooth* button as well, which makes the penguin really slick.

Note: you may see some weird effects at the bottom and the tail after subsurfing the penguin. If so, there is an issue with normals: they have to

be all pointing outwards. This can be achieved by selecting all vertices in Edit Mode and recalculating the normals outside (**CTRL + N**). Click on the message to confirm. Note that **CTRL + SHIFT + N** will turn the normals inwards and that **W → Flip Normals** flips them.

Question: The finished penguin looks fine in Object Mode, but when I render it, it looks odd. Quite patchy.

Answer: Make sure you adjust the “Render Levels” parameter (directly under “Levels”) to be greater-or-equal than “Levels”.

2.28.10 Extra

The penguin can be colored or textured, but that will be part of later tutorials!



- This is what the penguin (sans tail) looks like, textured and ready. Orbisonitrum



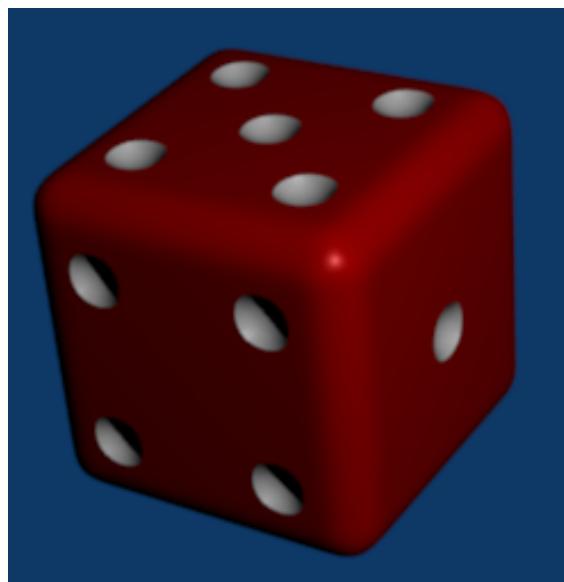
- The eyes are there, just not easily visible in the thumb. At the top of the white part, two faces on the chest were subdivided to give the white more of a curve at the top. The faces were selected that were going to be white, and the I used separate (**P**) to make them a different mesh. I used a white material for the chest, black for the body, and grey uvspheres for the eyes. Apparently, an easier way to colour the chest can be found at [Multiple Materials](#) please feel free to replace this with your own image of the penguin you made, with comments on how you put your own style into it



- A pretty basic picture of a penguin. I subdivided the stomach and eyes, but then I also added some eyeballs by making a UVSphere, cutting the top of it off, and then placing it inside of my penguin's head. All the colors have

specular colors, giving the penguin a slight blue glow under the black.

2.29 Dicing With Depth (Dice Modeling)



Dice are objects which are familiar to us all, used in countless games. Here we will model how to make a single *die* (or a single dice, if you prefer).

The end result should look something like at right. Notice the following features:

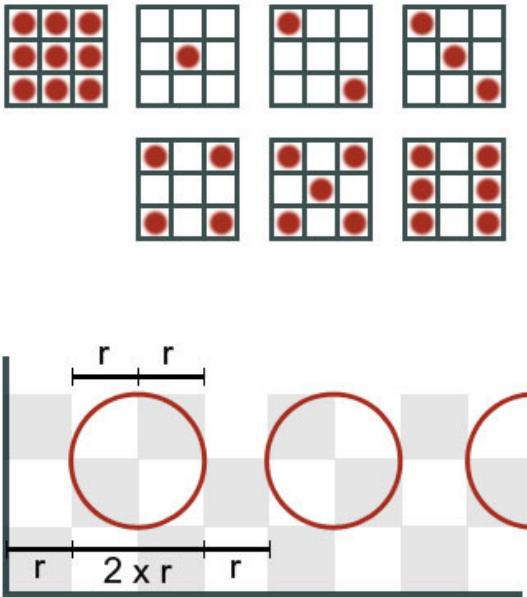
- different colours for the pips versus the body of the die
- rounded edges and corners, while the faces are (mostly) flat
- round, recessed pips.

Achieving all these effects will be a good exercise of your mesh-modelling skills.

Characteristics of the Die

The number of pips on each face of a die ranges from 1 to 6. However, in order to space them correctly, there needs to be 9 positions for a pip on each face, even though not all of them will be filled on any face.

We will also proportion our die so that the diameter of each pip will be twice the horizontal or vertical distance between pips, even though if you look at the illustration the pips on the finished die look smaller because they eventually do not occupy this full diameter. As you will

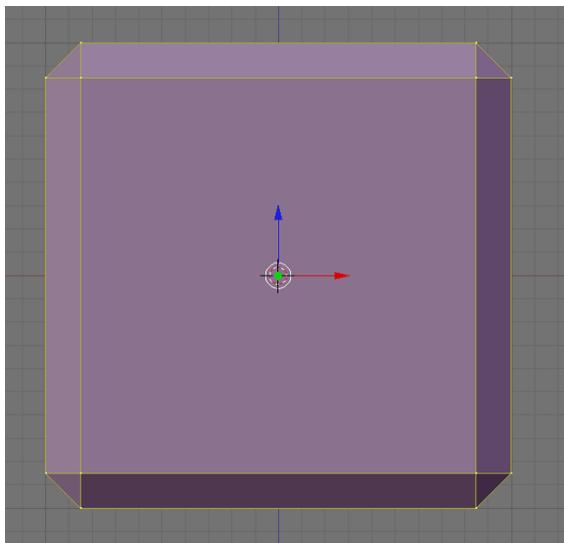


see, this makes it easy to get the proportions uniform with a relatively small number of *loop cuts*.

Also note one of the characteristics of real dice is that the number of pips on opposite sides adds up to 7.

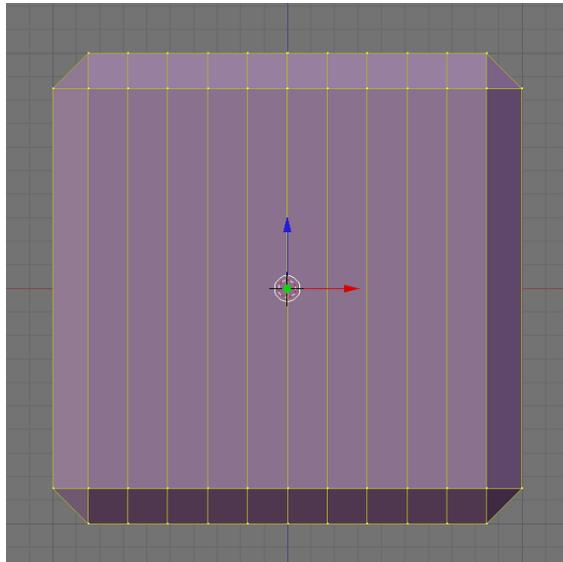
2.29.1 The Basic Mesh

Open a new Blender document. Select RMB the default cube, and TAB into Edit mode.



First of all, let's apply a small amount of bevel to the corners and edges: with *all* the vertices selected, press CTRL + B to start bevelling—not too much, press LMB when you see something like at right.

Now, we need to make a bunch of loop cuts to mark out

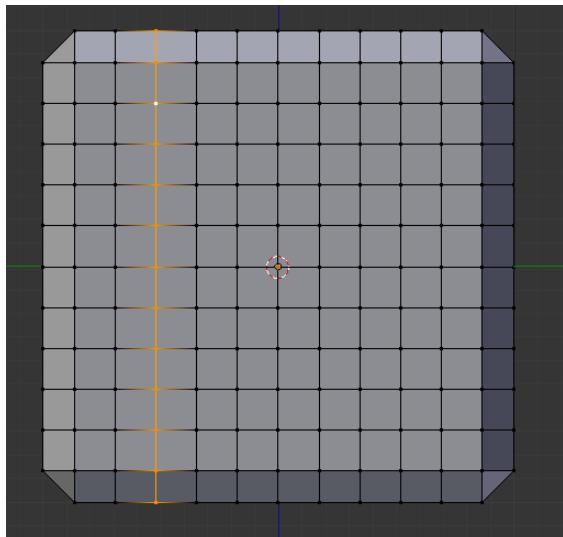


The first cut is the deepest

the areas where the pips will go. To simplify things, we will do cuts through the middle of each pip, as well as between pips—a total of 9 loops each way—then go back and remove the unneeded cuts. That way we get the 2:1 proportions correct with a minimum of effort. Who says it doesn't pay to plan ahead?

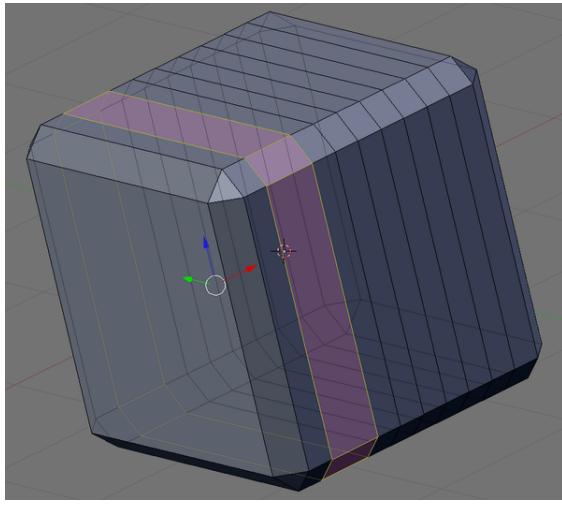
To make a loop cut, press CTRL + R , then move the cursor over the edge you wish to subdivide: you should see a single magenta loop appear. Now press 9KEY , and you should see this change to 9 loops. Next, click LMB to change the lines to yellow and really start cutting, then immediately press RMB to finish the cut *without* moving the cuts from their initial positions.

You will need to do this 3 times, between the 3 pairs of opposing faces.

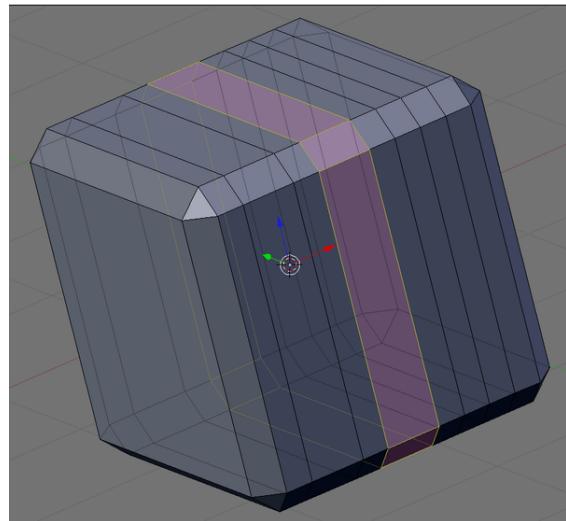


First loop to remove selected

Now look at each face of your cube: you will see an



First loop gone

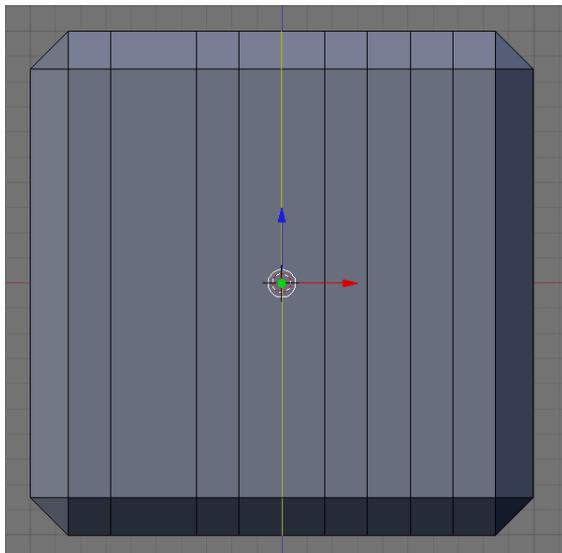


Middle loop gone

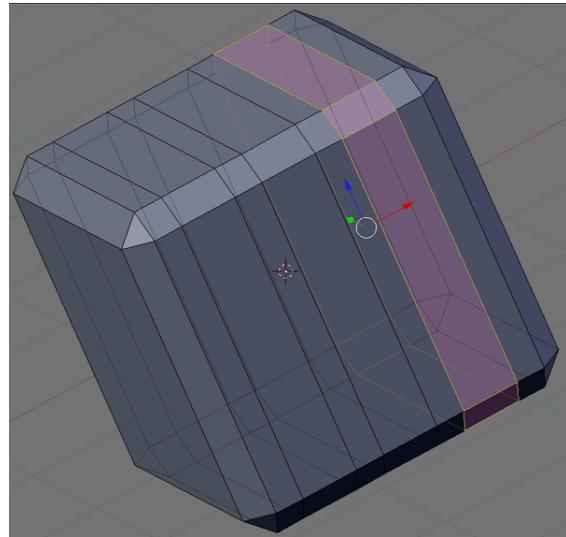
11×11 grid of vertices, with the corner vertices each adjoining a triangle at the corner. Make sure nothing is initially selected. Count the *third* vertex in from a corner, and move the mouse slightly so it is over the edge leading from this vertex into the interior of the face, just to avoid confusion; now Shift + ALT + RMB, and you should select a loop like at right.

If you get the wrong selection, just A to clear it, and try again. Remember to click over an *edge* that is running the right way, rather than a vertex, otherwise Blender is liable to select a loop running the wrong way.

Press DEL or X, and in the deletion menu that appears, select “Edge Loops”. The selected loop should disappear.



Middle loop selected



Third loop selected

As before, you’ll need to do this deletion of 3 loops 3 times, between the 3 pairs of opposing faces. When you are finished, the resulting mesh should look like at right.

2.29.2 Making the Pips

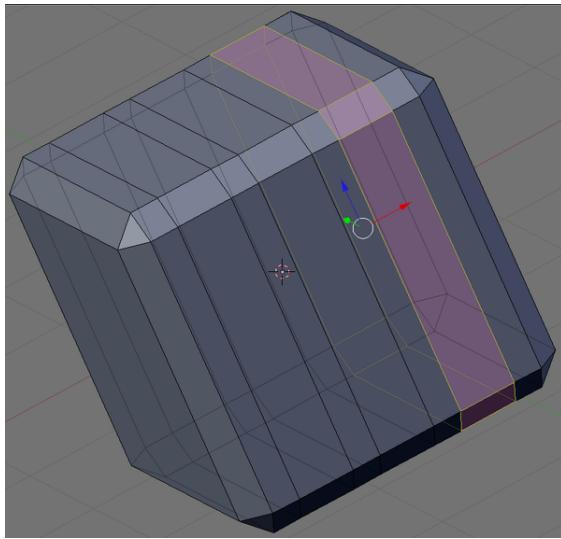
Now we will hollow out the spots or pips. The idea is to *extrude* selected ones of the large squares on each face *inwards* into the cube (“*intrude*” rather than “*extrude*”, perhaps?), then split the bottom of each resulting pit into four triangles each by adding an extra vertex in the middle; then pushing in this vertex will give us the hollow for the pip (as at right).

We will show three methods for doing this.

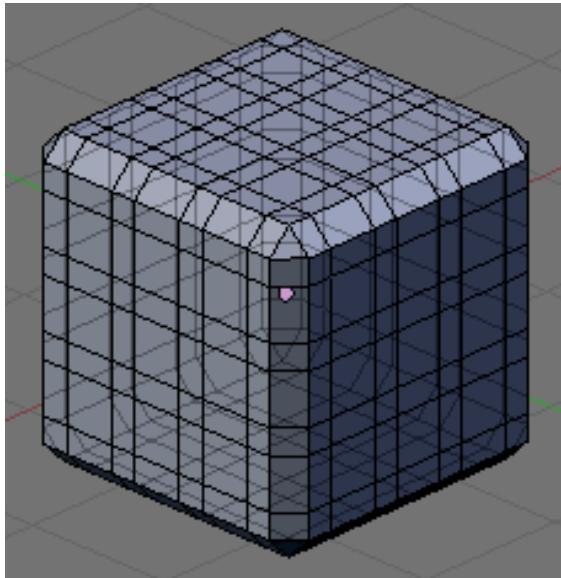
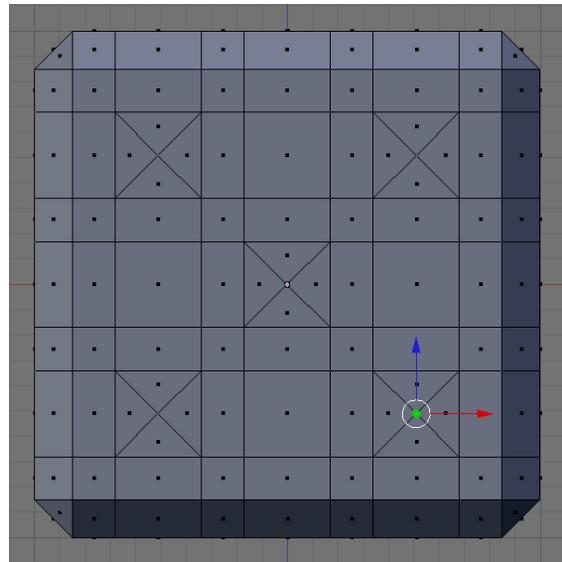
Do this individually for each face that is to be made into a pip.

Going in from the edge of the face, skip over one loop, and remove the middle one.

Then finally remove the third one in from the opposite edge.



Third loop gone



Method 1 (Slowest)

Select all the large squares to be made into pips (perhaps do one entire face at a time). Press **CTRL + T** to triangulate these faces. Now switch to edge-select mode, and select the new diagonal edge that has appeared in the middle of the original squares; bring up **CTRL + E** for edge Specials and select Subdivide (leave the number of subdivisions at the default 1). Things won't look any different, but each triangle is now a quad, with an extra vertex added to the middle of each of those diagonals.

Now switch back to face-select mode, select those subdivided halves of all the squares, and **CTRL + T** to triangulate again.

Method 2 (Faster)

Switch to face-select mode. Select all the large squares to be made into pips (perhaps do one entire face at a time). Do **E** to extrude, and immediately press **ESC** to leave the new extruded vertices in the same positions as the old ones; now do **ALT + M** to merge and select the “Collapse” option.

Method 3 (Fastest)

Select the relevant large squares on each side to make the required numbers for each face of a real die. From **CTRL + F** for face Specials, select “Poke Faces” (or more directly, do **ALT + P**). Leave the settings at their defaults (in particular, leave the Poke Offset at 0).

Digging The Holes

Now that you have crossed the faces of your pips, select the middle vertices one side at a time, and **G** rab inwards by a distance of 0.17 (make sure you do it at the appropriate axis, pres **Z , X , Y**) along the appropriate axis.

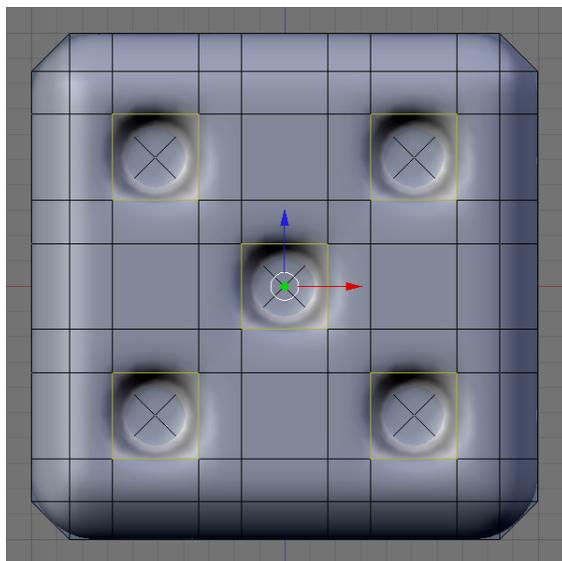
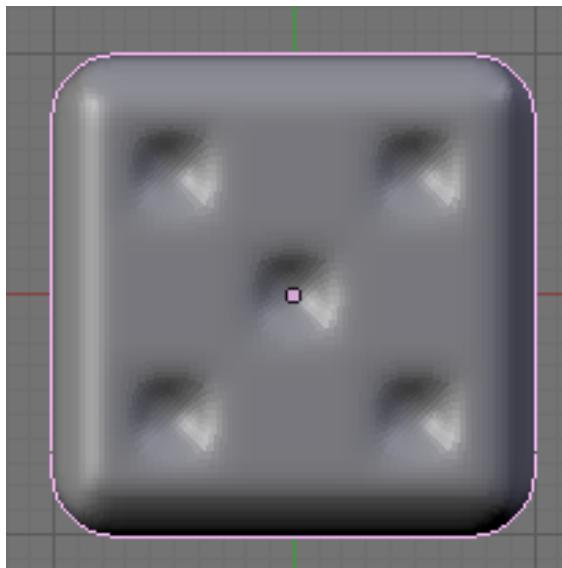
Noob Note: Do this side after side, otherwise you will translate all your vertices to one side and not all inside.

2.29.3 Rounding Things Off

Now apply a subsurf modifier and set the shading to smooth, the result should look something like this.

The trouble is, the pips look too much like dimples rather than proper holes. To fix this, we can apply a *crease* to sharpen up the edges.

In edge-select mode, select the four edges around the opening of each pip. Press **SHIFT + E**, and move the



mouse until the edges take on the darkest magenta tint. That should firm up the edges of the holes a bit.

2.29.4 Colouring Your Die

First, go to the Materials Context  and set up the colour for the main part of the die (say, make it red as in the illustration above). This can be done in Object or Edit modes.

Next, click the “+” button to add a second, initially empty, material slot, then click the “+New” button that appears below it to create a new material for this slot. Make this material white, for the pips. Then, select the pips themselves, and use the “Assign” button to colour them with this material as detailed below.

Quickly Selecting The Pips, Method 1

Go to the Select menu, and select the option “Select Faces by Sides”. A panel will appear at the bottom of the Tool Shelf (use T to toggle the visibility of this), with a field titled “Number of Vertices”, initially showing 4; change this to 3, and it should select all the triangles, which includes the pips as well as the corners of the cube. You can deselect the latter one by one by SHIFT + RMB ing them.

Quickly Selecting The Pips, Method 2

Alternatively, you can do just select the faces of each pip.

Quickly Selecting The Pips, Method 3 (Fastest)

(Works in blender 2.6, not sure about earlier versions.) In face select mode select one of the triangular faces of one the pips. Hit space and type “select similar” then choose “area.” If this selects too many faces along the edge of the die, cancel and try again with “perimeter.”

2.30 Model a Goblet

Why a goblet? After all, goblets (fancy containers for consumable liquids) are not really all that important in the real world. However, they represent a very large class of interesting objects: those that are radially symmetric. Also, they are fun to play with.

To model a goblet and actually see the result, we do the following steps:

- create the actual mesh of the object.
- apply a *material* to the object.
- create and light a scene to display the goblet
- render the scene.

The following tutorials demonstrate three different approaches to creating the mesh model, and then show two different materials (glass and silver) that may be applied to the goblet. Finally, we add the minimal elements to the scene to permit an “interesting” rendering.

2.30.1 Three ways to build the mesh

As a beginner, you may wonder why we describe three different ways to build the mesh. After all, surely the Pros know of a “best” method?! The answer is that there is not really a best method. Blender provides a sophisticated toolkit, and different blender artists will become

familiar with different tools. If, after you gain experience, you become comfortable with cube extrusion, then this may become your preferred tool. If you need more than four vertices per “circle” you may find that cylinder extrusion is better for some particular object. If you are comfortable with “spinning” an object, then spinning may be right for you. All three techniques generate an object that is defined by a set of vertices on a set of circles that are “stacked” on a common axis.

After we build the mesh using one of the three methods, we can apply any number of fancy techniques for texture and rendering. We provide two simple “cookbook” approaches in this section so you can see the result, but materials are treated much more extensively in other sections of the book.

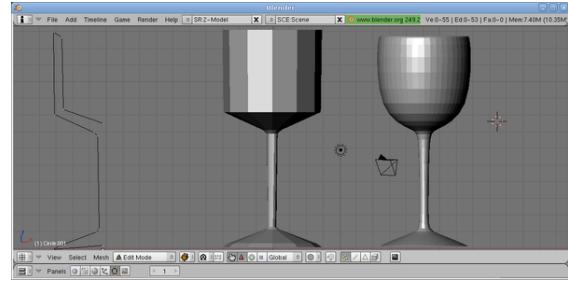
In the next three sections of the book we describe the “cube extrusion” approach, the “cylinder” approach, and the “spin” approach. Following sections describe three textures that can be applied to objects that are created with any of the approaches, and the final tutorial provides a simple way to render the result.

The “cube extrusion” approach is a basic technique that all blender artists will become familiar with, to model most types of objects. Since it is such a basic technique, it may become the most efficient way for you to create objects, and you may choose to use it wherever possible rather than switching to a less familiar technique. You can see from the tutorial that this technique is perfectly acceptable for goblets, and by extension it will work for similar objects.

You may prefer the “cylinder extrusion” technique when you need more than four vertices per circle, because cylinder extrusion is very similar to cube extrusion. Thus your proficiency with cube extrusion will directly carry over to cylinder extrusion.

“Spinning” is conceptually different, even though it creates the same mesh as cylinder extrusion. Spinning is the easiest way to create a radially symmetric model from a two-dimensional description of the cross-section of an object.

2.30.2 How many circles?



Each of the three methods defines a set of stacked circles, with vertices equidistant around each circle. The “cube extrusion” technique has four vertices on each circle, while the cylinder and spin techniques have a user-specified number of vertices per circle. But how did we pick the number of circles and the spacing between them?

The answer is that we know in advance that we intend to use a technique called “subsurf”. Subsurf interpolates additional vertices between those we explicitly specify, and this “smooths out” the profile of our goblet.

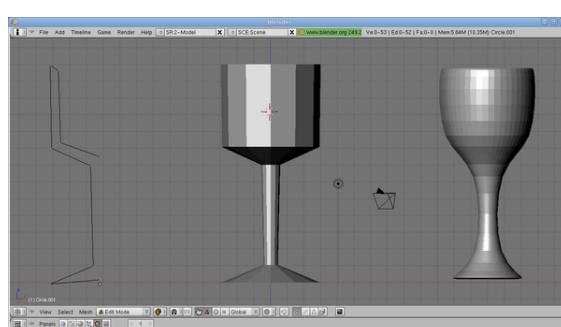
This will cause problems where we need a fairly sharp “bend” in the profile, as between the base and the stem and between the stem and the “bulb” of the goblet. At each point in the profile that has a (fairly) sharp bend, we need for the vertical circles to be close to each other. The two images illustrate this effect. Each image shows an outline of the object, the spun object, and the result after subsurf (level 2) has been applied. The outlines and the spun images are nearly identical, but the subsurfed result is very different, because there are three extra vertices in the second outline: one just above the bottom of the base, one at the top of the base, and one below the bulb: these are the three locations where we need relatively sharp changes in contour. When we spin the outline, the first outline generates seven circles and two degenerate circles, while the second outline generates ten circles and two degenerate circles. Those extra three circles make all the difference. The same phenomenon occurs with the other two modelling techniques.

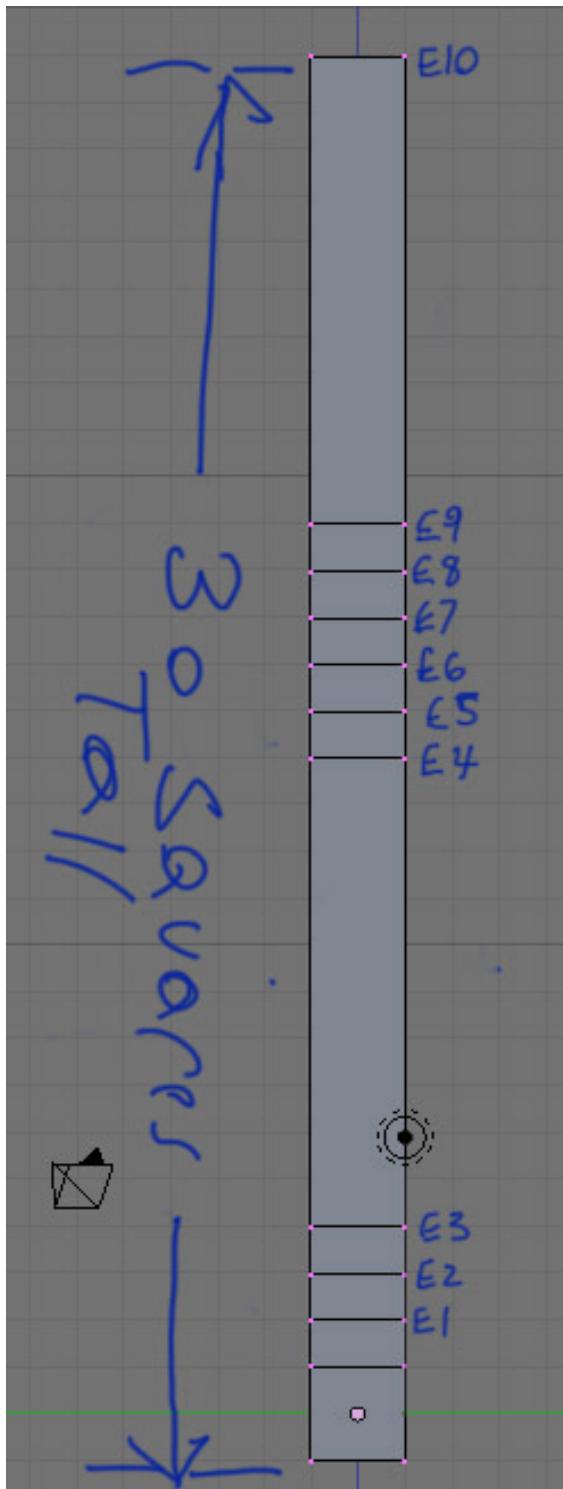
2.31 Model a Silver Goblet

2.31.1 Basic Shape

At first glance, the goblet looks like it is composed of cylinders. However, while it is possible to model the goblet with a cylinder mesh, it is easier to make a goblet by using cubes. Cubes make the goblet faster to make and it makes fewer vertices to track. Now, lets start making the basic shape of the goblet.

Start with the default cube and go to Edit Mode (TAB). Toggle off “limit selection to visible”. Move to the side view (NUM3). Toggle to ortho view. Box select (B) the top edges of the cube. Extrude (E) upward about



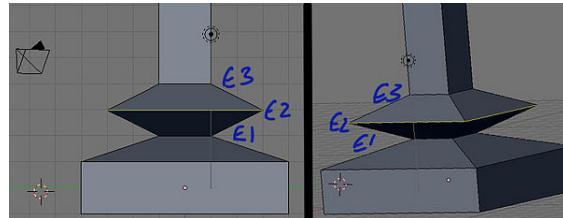


one grid square. Hold down CTRL while extruding for incremental movement; by default movement should be restricted to the Z-axis (normal to the selection); if necessary, press Z to toggle this effect. This extrusion is called E1. Repeat 2 more times for extrusion 2 and 3 (E2 and E3).

Now Extrude a longer piece upwards, for the Goblet's stem, of 10 grid squares (E4). Next we will define the

area for the 2 top knobs and the bottom of the glass by Extruding upwards 5 more times at 1 grid square each (E5-E9). Next do another upward Extrusion of about 10 grid squares (E10). This is the actual glass itself.

Inflate the Glass



Sizing the first knob

Now, let's begin to inflate the glass. First, clear all selections by hitting the A. Make sure you are still in the side view (NUM3). Box select (B) the bottom most cube (all 8 vertices, not just the bottom 4). Then expand it outward by scaling (S), then pressing SHIFT + Z to lock/prevent any scaling in height along the Z-axis and finally pressing 4 to quadruple its size. Next, deselect all vertices (A), select (B) the 4th pair of vertices (E2), scale them (S), lock scaling (SHIFT + Z) and triple the vertices in size (S and 3).

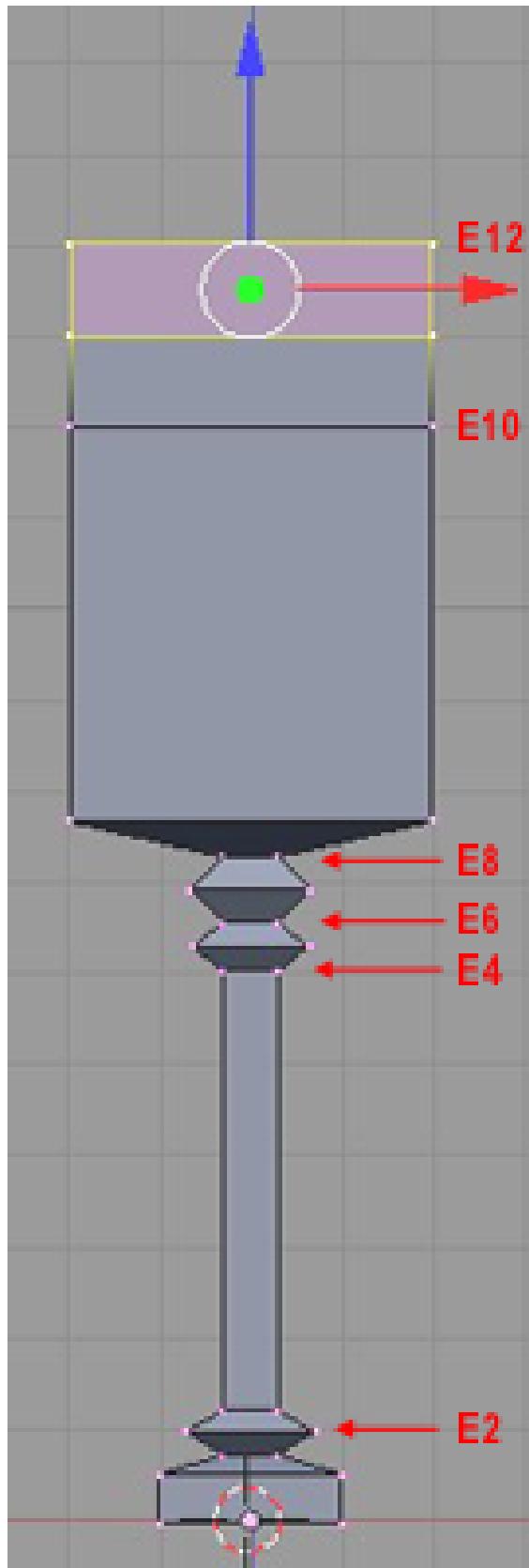
Do the same for E5 and E7 (the 7th and 9th pairs of vertices, respectively). Then expand (S and SHIFT + Z) the top, Goblet rectangle (E9 and E10) to 6 times its current size (6).

Make the cup's interior

Last is the cup's interior. Once again, you should still be in the side view (NUM3). Box select (B) the very top most vertices of the Goblet (E10). Once selected, get a better view by changing to 'Orbit Up' (NUM8). With the top surface selected, initiate an Extrusion (E), followed by a termination with the ESCAPE . It will appear as if nothing has happened, but new, overlapping edges have been created and are now selected (this also creates E11). Next Scale (S) the selected vertices to 90% of the original size (.9). This creates the inside lip of the cup. Now, Extrude (E) the interior lip, along the Z-axis downward (-10) to create the bottom interior of the Goblet (E12). Finish off by selecting all, and then a 'Remove Doubles' (W → Remove Doubles), for good measure. It should look something like this.

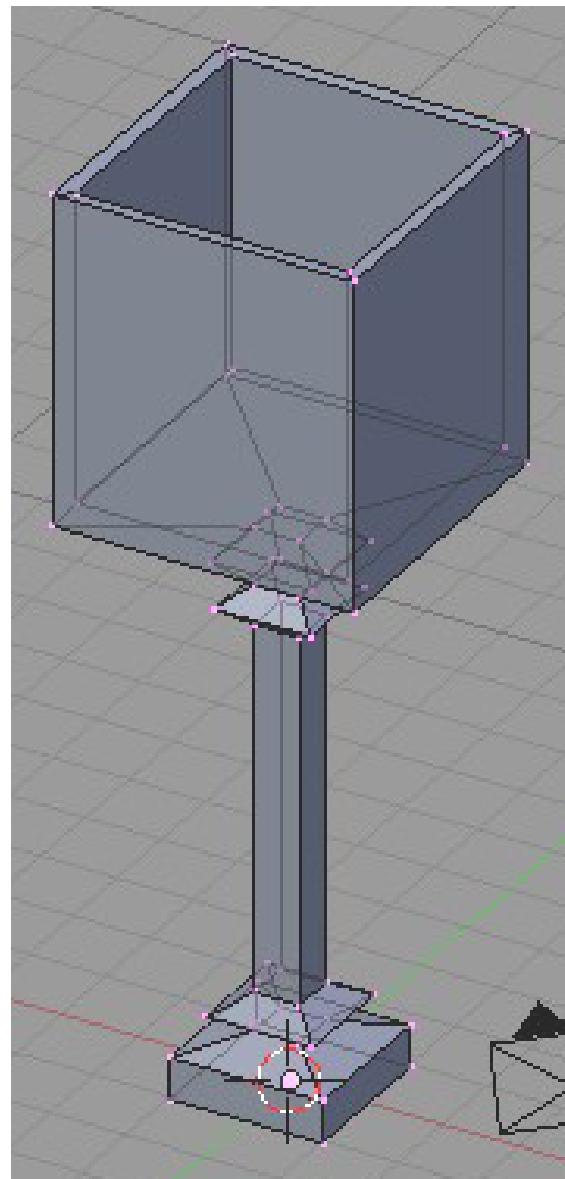
2.31.2 Smoothing and Defining

Time to take the mesh and turn it into a proper goblet. Add a subsurf modifier with 3 view subdivisions to the mesh. Change to Object Mode (TAB) and select smooth



All extrusions

shading. The cube-looking mesh will now look like an object that was created from a cylinder. This has removed



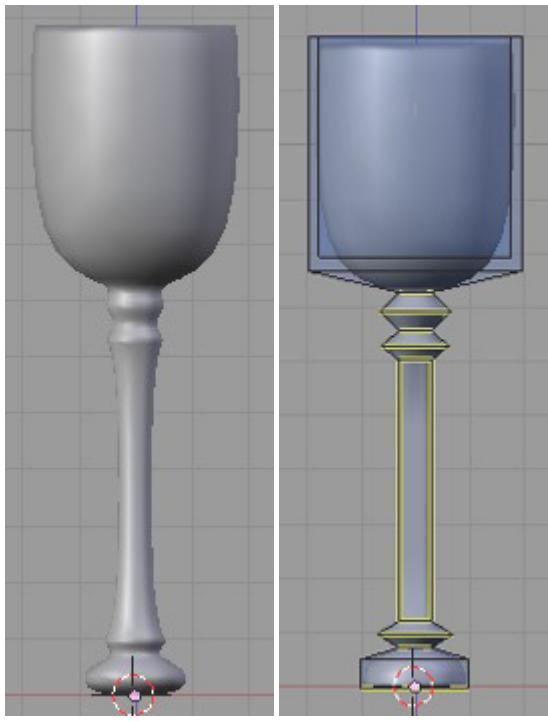
View of the finished basic goblet

all our crisp edges, but our goblet is looking very unstable! Let's rectify things by flattening the bottom.

Noob Note: If your goblet has a bulge in it after applying the subsurf, go to **Edit mode** (Tab), select all (A), press W , and select “Remove doubles”. (If a menu for boolean operations appears when you hit the W , then you’re not in **Edit mode**; change immediately to **Edit mode** and try the W again.)

Select the four edges that surround the small circle at the very bottom (the very lowest set of edges) and press SHIFT + E to enable creasing. Now drag the mouse up and down to select the level of the crease. When you’re satisfied, hit LMB . Repeat the process for other edges you want to be sharp. I’ve turned on Draw Creases under **Mesh Tools 1** to illustrate which edges have been creased (highlighted yellow) in this example.

That concludes the creation of the goblet. Save the scene for use in the lighting tutorial.



2.31.3 Quick links for the impatient

To jump to the relevant lighting section, go to [Blender 3D: Noob to Pro/Light a Silver Goblet](#).

What about applying glass look already? Take a sneak peek at [Blender 3D: Noob to Pro/Spin a goblet#Material](#).

2.32 Model a Silver Goblet Another Way

2.32.1 Preliminaries

This is a version of “Model a Silver Goblet” but starting from a cylinder rather than a cube.

Start Blender or start a new scene (**Ctrl + X**) and delete the default cube.

Change to Top view (**Num7**), make sure you're in Object mode and add a cylinder (**Shift + A Mesh > Cylinder**): in the popup menu, change the number of vertices from 32 to 16, the Radius from 1.000 to 1.800, the Depth from 2.000 to 0.100 and leave “Cap Ends” as it is.

Change to Front view (**Num1**).

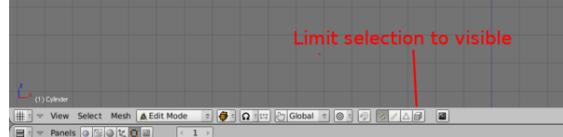
2.32.2 Description of the modeling steps

The steps in this tutorial are almost all made up of “extrude” and “scale”: so, to avoid repeating key sequences

every time,

extrude means

1. make sure “Limit selection to visible” is off



Shows “Limit selection to visible” icon

2. box select the top vertices of the cylinder: press **B** , click and drag LMB to make a rectangle around the top vertices
3. press **E** , press **Z** and type in the amount of the extrusion and press Enter ; you can move the mouse instead but it is quicker and easier to type it in.

For example, the following keystroke sequence extrudes by 1.5:

B , click and drag LMB to make a rectangle around the top vertices; **E Z 1 . 5 Enter** .

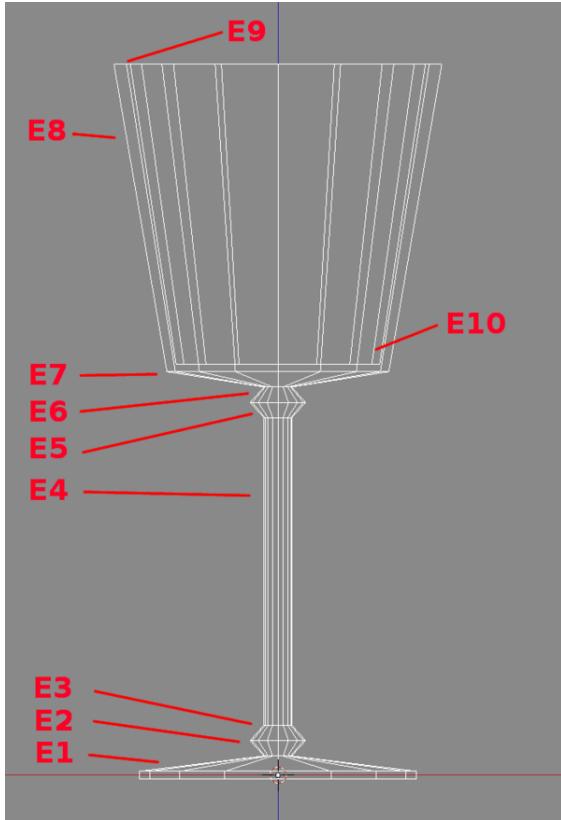
scale scaling is by default restricted to the X-Y plane; although the numbers in the bottom left corner of the 3Dview show Z changing, in fact only X and Y change, and by equal amounts.

1. press **S** , type in the value and press enter — you can use the mouse instead but it is quicker and easier to type in the number.

2.32.3 Creating the Goblet

This diagram shows the connection between the E-numbers and the goblet construction.

- **E1:** Deselect all vertices (**A**), Box-select the top vertices (**B-key**), and extrude by 0.2: you may need to zoom in (**SCROLL**) to do this as it's quite thin. Scale to 0.1.
- **E2:** Extrude by 0.2, scale by 2.
- **E3:** Extrude by 0.2, scale by 0.5 to make the lower knob.
- **E4:** Extrude by 4 to make the stem.
- **E5:** Extrude by 0.2, scale by 2.
- **E6:** Extrude by 0.2, scale by 0.5 to make the upper knob.
- **E7:** Extrude by 0.2, scale by 8 to make the base of the cup.



E-numbers and construction steps

- **E8:** Extrude by 4, if you wish to make a flared cup, you can scale by 1.5.

- **E9:** Extrude by 0.0, scale by 0.9 to make the rim of the cup. (This will create a new ring of vertices and then move them in towards the centre.)

Now go into Wireframe mode (Z) so you can see inside to guide the next few steps.

- **E10:** Extrude by -3.9 , that is, downwards, and scale by 0.69: you can do this last scaling with the mouse, if you like, to get the edges of the inside of the cup and the outside parallel.

- **E11:** Extrude by 0.0, scale by 0.0 to make the inside of the cup. Press W Remove Doubles to merge the centre vertexes.

You now have a goblet, the base of the inside of the cup is the face of the last extrusion, is circular and flat as it derives from a cylinder.

If you haven't already saved your work-in-progress, now would be a good time.

2.32.4 Subsurfing and smoothing the goblet

The last step is to subsurf and smooth: go into Object mode and enable Solid mode again (Z).

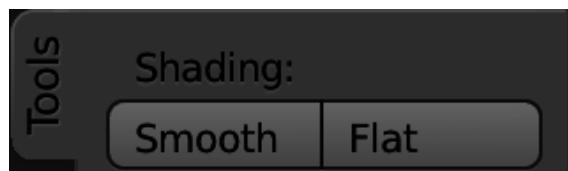
Select the Editing panel from the Buttons window (F9) and, in the “Modifiers” panel, click on “Add Modifier”, select “Subdivision Surface” from the popup menu and, in the Subsurf display, increase “View subdivisions” from 1 to 2.



Shows the Editing panel icon and the Modifier panel

At the bottom right of the “Links and Materials” panel, click on the “Shading: Smooth” button. At the bottom of the cup you will see fluting — this is an artifact caused by smoothing and subsurfing triangles on a curved surface. Here it adds to the appearance, don't you think?

In Blender v2.78, the “Smooth” button is located in the “Tools” menu under the “Edit” tab.



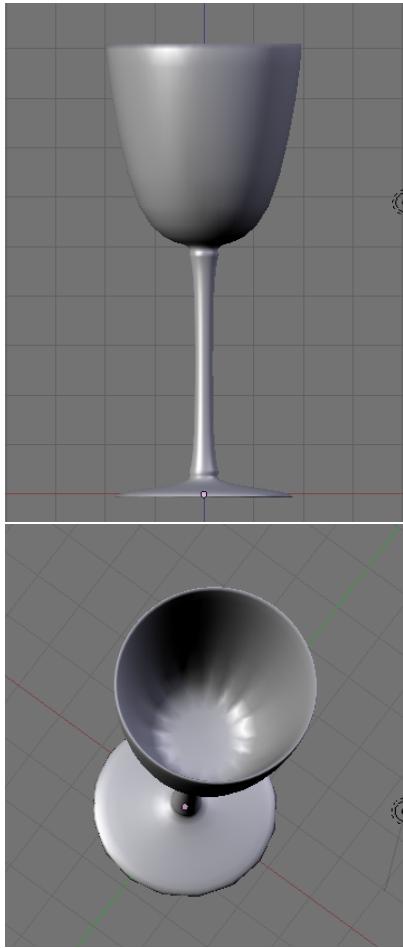
Tools shading

2.32.5 Flattening the base of the goblet

The base of the goblet is curved due to the subsurfing, so needs to be flattened.

Go into Edit mode, deselect all, box-select the lowest set of vertices, then crease (remove the subsurfing) by pressing Shift + E then 1 enter .

The final result should look something like this:



Save the scene for use in the lighting tutorial. To jump to the relevant lighting section, go to [Blender 3D: Noob to Pro/Light a Silver Goblet](#)



Photo of a real-world goblet

2.33 Spin a goblet

The **spin** technique is a good choice when you want to model an object that is radially symmetric and you know what the cross section of the object looks like. This is the virtual equivalent of using a **lathe** to create an object in the real world. With the spin technique, we draw an outline of one half of the outline of the object, and then spin the outline about an axis to create the object's mesh.

2.33.1 Modeling

Here is a real-world goblet. The picture is not an orthographic image, so we cannot directly copy the outline, but it is close.

Setup

We now model the outline. Let's start with the default model. We want to create an initial object consisting of a two-dimensional outline, so select the cube, go to edit

mode, and delete all the vertices. This leaves us with an object that has no vertices as a nice place to start. We want the resulting goblet to sit at the origin with the Z axis as its axis of symmetry so we get a front view (press NUM1). Since we will be working directly with mesh vertices, the *manipulator* is a hindrance, so turn it off (press Ctrl + Space). (With the manipulator on, it is easy to accidentally move a vertex out of the editing plane, but we want a 2D cross-section.) We are now ready to create the two-dimensional outline as a chain of vertices with the first and last vertices on the axis.

First Vertex

Now, place the first vertex: this will be the center of the bottom of the goblet, so place it slightly above the origin on the Z axis (Ctrl + LMB). Why? Well, for two reasons: in the real world, the bottom of a goblet is not actually flat. Instead, the rim is lower than the center of the bottom, so the bottom is concave and the goblet sits on a flat surface without wobbling. The second reason is that we intend

to use the “subsurf” technique, and this technique will make a flat surface slightly convex in our virtual world, so we will preemptively start the bottom of the base with a slightly concave surface.

Finish the outline

Add the second vertex, which will be on the rim of the goblet’s base: move the cursor to the X axis at about -3 , and add a vertex (press **Ctrl + LMB**, or **E**, **Enter**). Add additional vertices to your outline by moving the cursor to the desired location and adding,^[1] for as many vertices as you need to accurately model the outline. Since you will be using subsurf later, make sure that you place two vertices near each other when you need a sharp curve in your outline. Otherwise, subsurf will convert your sharp curve into a wide smooth curve.

Eventually, you will place the last vertex, which will become the point at the bottom of the inside of the goblet. This point should be on the Z axis. Your outline is done!

Spin

Now, to spin it. first, make sure that you are still in edit mode, and that the last vertex is selected. Set the cursor to the selection (**Shift + S** , then *cursor→selection*). Now, get a top view (**NUM7**): you are now looking at your outline from the top, and it should look like a straight line along the X axis with one end at the cursor on the Z axis. (If this is not the case, select and move points to the X axis and check your work by switching back to the front view, then come back to the top view.) Now select the whole outline and then move to the button menu to perform the spin (or use the Spin option under Mesh Tools, Add, in the Toolshelf). Set the rotation to 360 and the steps to 12 (or another number of your choice). An elaborate circle will appear. Go to the front view (**NUM1**) to see your un-smooth goblet.

Finish up

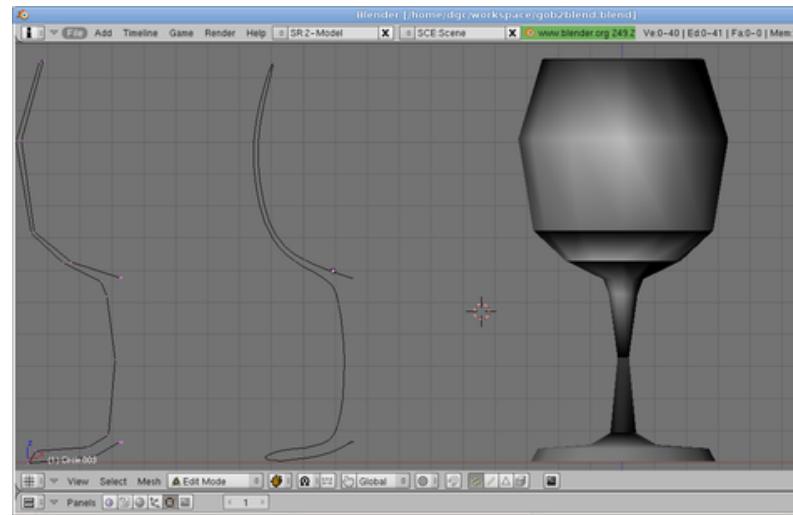
Since you are in edit mode and you have the whole mesh selected, this is a good time to remove duplicate vertices. The two vertices on the Z axis (base bottom center and bulb bottom center) were duplicated 12 times, and the entire outline was duplicated once when the circle closed at the end of the spin. Remove the duplicates (press **W** and select *Remove Doubles*). If you fail to do this, the subsurf operation will create a cusp at the two centers, and a crease at the duplicated outline.

If you placed the two Z-axis vertices by hand, they may not be exactly on the axis, and therefore may still be duplicated, leaving a tiny “hole” at the axis. Fix this by merging each of the two sets separately:

1. select the vertices to merge.

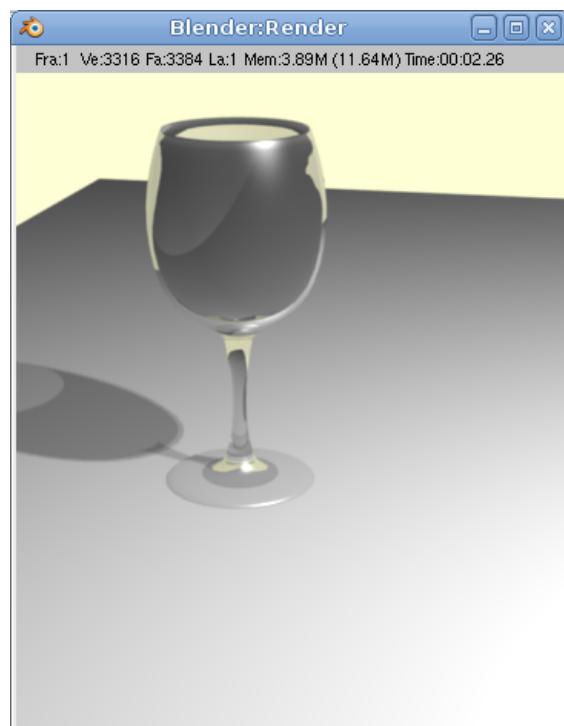
2. merge at center (press **W** ,*Merge → At Center*, **Enter**).

Now start to smooth by using the subsurf in the button window. First add a modifier and select subsurf. Set levels to 2 Then, click on *smooth* in the Tool shelf under “Tools” -> “Shading”. You now have a goblet model.



The image shows four objects: the two-dimensional outline, a smoothed version of the two-dimensional outline, the result of spinning the outline, and the result of subsurfing and smoothing. As you follow the procedure above, you will not actually have more than one object at a time as shown here.

2.33.2 Rendering



As with the other two goblets, It is difficult to fully evaluate the model unless you render it. However, a “pretty” rendering requires at least minimal materials, lighting, and scenery. This section is a cookbook approach to providing these minimal elements and is not really a useful tutorial, so we won’t explain the concepts. These topics are treated at length in later tutorials. If you wish to explore these subjects in more depth, go to the appropriate tutorials.

- The Goblet material: plain glass.
- The tablecloth: A (non-existent) table with a white tablecloth, in a featureless room painted yellow.
- The lighting: one bright lamp.

Material

- Go to object mode and select the goblet.
- In the Materials button window, add a material to the goblet, and name it “glass”.
- In the *Material* tab, change the color in *Diffuse* section to black (set R, G, and B all to 0.000). Here, “black” merely means “do not add any color”. It does not mean that the goblet looks black.
- In *Transparency* section, pick Raytrace. Set Alpha to 0.1 (i.e., quite transparent), set IOR to 1.5, and set Depth to 6 (or higher on a very fast computer).

Tablecloth and room

- In the main window, put the cursor at the origin (Center) and switch to top view. In object mode, add a plane and then scale it to quite large. This is your tablecloth
- Go to object mode and select the tablecloth.
- In the *Material* tab, set the color to white and ensure the Alpha is 1.000 (i.e., opaque.)
- In the *Shadow* section, check the Receive Transparent checkbox. (Older versions: in the *Shaders* tab, click on Trashado.) This allows the tablecloth to show the ray traced shadow of your goblet instead of a fake shadow.
- To paint the room, in the *World* tab check Blend Sky and change Zenith Color to e.g. yellow.

Lighting

If you started with the default camera and light the scene will be too dark and the shadow effect from the lamp will not be too pretty. To fix this:

- Move the lamp higher and farther away.
- Turn up the Energy.
- In the lamp’s *Shadow* section, pick Ray Shadow.

Camera

Now adjust the camera:

- Shift to camera view.
- Dolly and aim the camera.
- Move the camera back some.
- Render.

[1]

2.34 Light a Silver Goblet (Early look at lighting)

Note that the images are outdated.

2.34.1 Techniques

You should know how to:

- Perform actions discussed previously in the tutorial.

This section will recap or introduce:

- Reflective material
- Positioning camera and light
- Editing the World colors

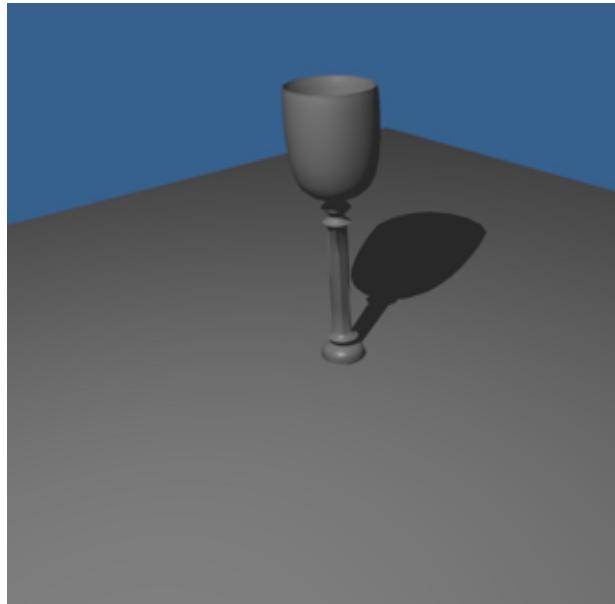
2.34.2 Objects in the Scene

Create the goblet discussed in [Model a Silver Goblet](#) or load it if previously made. If you haven’t already made the goblet, feel free to try the tutorial using a sphere or something else instead and you will still get a good outcome. In Object Mode with **NUM 7** view, add a plane mesh. Scale the plane to a very large size and make sure the goblet is sitting comfortably on top of it.

Select the camera and move it so that the goblet, and its reflection in the plane will be seen or else if you want. You can see the numerical location of the camera by bringing up the *Transform Properties* window by pressing **NKEY** in the viewport. In my example where 0,0,0 is the bottom center of the goblet, the camera is located at 27, -21, 19 XYZ with a rotation of 63.5, 0.62, 46.7.

Create a Sun with **Shift+A** → Lamp → Sun. And (in the lamp properties “Object Data”) set “energy” to “0.5”, and place it above the goblet. Move it at around 80 points on the Z-axis. It is very important that you place the lamp on the right spot cause it will give your goblet a 100 times more true to nature when you will give your goblet a silver texture. You can try placing a point or another lamp but it’s very difficult to get a realistic image then. If you choose a different lamp click on the World button in the “Properties” header (the section where you can edit the sky). Check the Environmental Lighting box. Set energy to “0.800”.

The rendering of this scene yields:



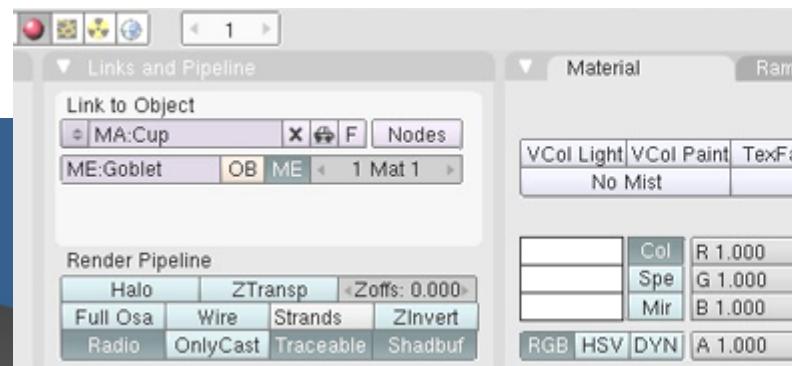
Old Picture.

2.34.3 Adding the Atmosphere

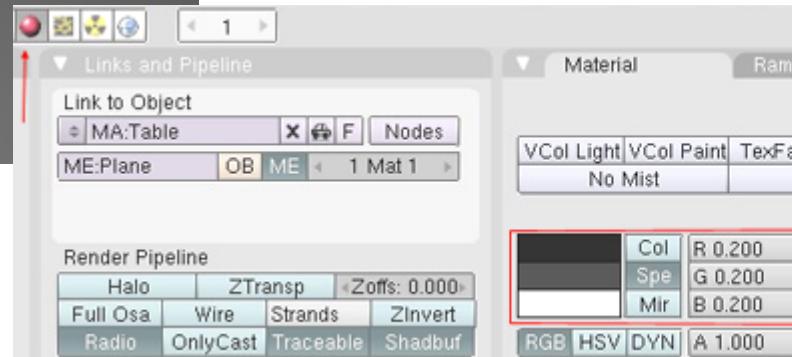
In Object Mode, select the goblet and go to the “Material Properties”. If no material is linked to the goblet, add new material. Rename the material ‘cup’ or something similar. The area of interest is the *Mirror* window. Highlighted below are the mirror options we’ll be playing with. Press the Mirror button to make the material act like a mirror and reflect light.

Move the “Reflectivity” slide to 0.85 or type it in after **LMB** on the number. This is how reflective the surface will be. A low number of 0.00 means that it reflects little while a high number of 1.00 reflects everything.

Also change the Fresnel slide from 0.0 to 1.4. This will increase the power of the Fresnel function. What this means is the color of the material will be strong because the light source is taken into consideration. If the Fresnel wasn’t used, the object would appear dark because the light source isn’t directly calculated in the mirror. Also, change the color of the goblet to white. Using a light color will give your goblet an interesting patina if you so choose.



Next, select the plane and modify the material, add if it is not there. We want the plane to be dark and shiny. Set *Diffuse* and *Specular* to near black for the color. For reflectivity, turn on *Mirror* to about 0.15 Reflective and ignore Fresnel this time.



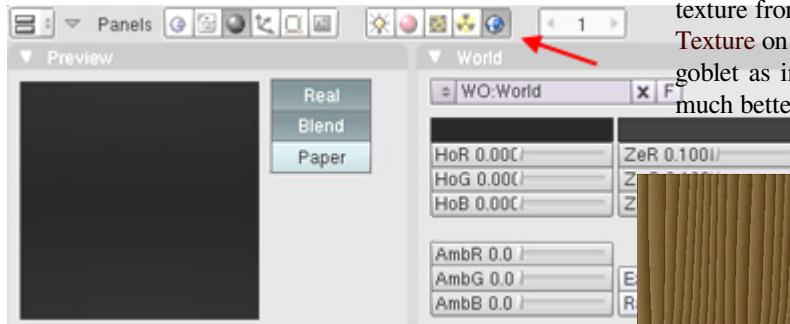
There is only one more thing to do before rendering the scene: change the world. Under the “Properties” Header is the “World” panel. Here you have *Paper Sky*, *Blend Sky*, and *Real Sky* buttons. There are also options for changing the color of the horizon (Horizon color), zenith (Zenith color), and ambient (Ambient color). We’re interested in these two windows at the moment.

Using *Real sky* and *Blend sky* will affect the way the horizon and zenith interact. Experiment with them to see what they do in the preview. In this example, *Real* and *Blend* are turned on.

The *Paper* button works a little differently in that what you see in the preview will essentially be the background of your render. This effect is most noticeable when your camera is rotated. Despite the camera rotation, the preview would still be ‘wallpapered’ on the render.

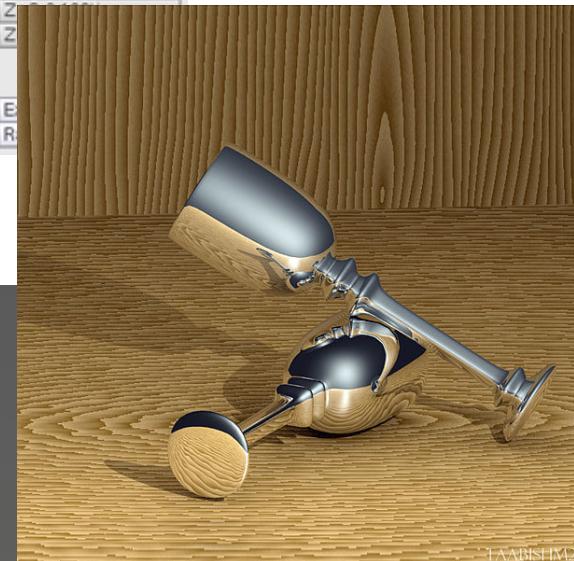
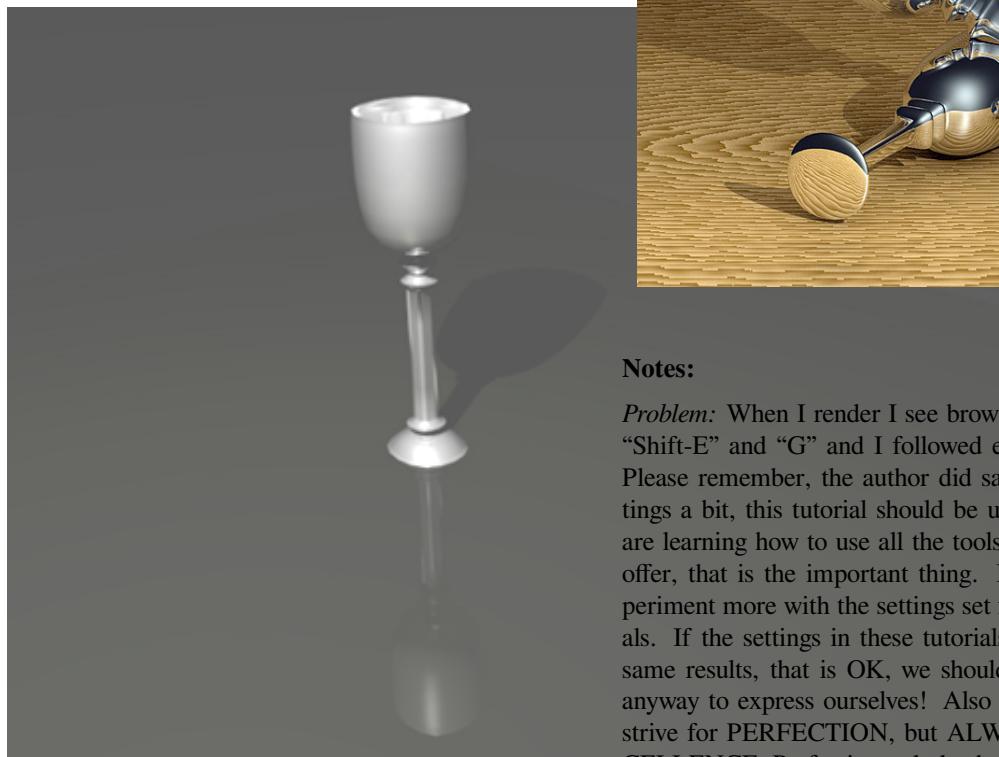
For our world, set the color close to black for the horizon,

zenith, and ambient.



4. (not compulsory but it may be required) Use the texture from this earlier tutorial: **Procedural Wood Texture** on the planes that are below and behind the goblet as in the picture below. It will give you a much better contrast.

That was the last step! Make sure the camera is in the right spot and render the scene. Here is the output of this example



Notes:

Problem: When I render I see brown where I have used “Shift-E” and “G” and I followed everything. *Answer:* Please remember, the author did say play with the settings a bit, this tutorial should be used as a guide. We are learning how to use all the tools that Blender has to offer, that is the important thing. It is up to us to experiment more with the settings set forth in these tutorials. If the settings in these tutorials do not give us the same results, that is OK, we should be changing them anyway to express ourselves! Also remember, NEVER strive for PERFECTION, but ALWAYS strive for EXCELLENCE. Perfection only leads to frustration, and it is frustrating enough, at times, to learn something new. Have fun learning, I know I am.

Noob Question: I managed to get it looking like the first picture above. How do I get it to look like the second?

Pro Answer: Change the reflection settings. a higher depth and a larger raymir value will make the goblet more “mirror” like as in the 2nd picture. the other settings should be left alone, or you can experiment with them to achieve the effect that you want. lighting is also important. the object that is to be reflected has to be illuminated as well as the object that is doing the illuminating. Different lights (don't use a hemi if you want it to be realistic) at different angles will give you a more realistic effect.

Noob Note: On the answer above, I didn't manage to do it with any of the things the pro said in the answer here, I found out that the key is to change the color of the material which is white (or close to white) in the upper picture.

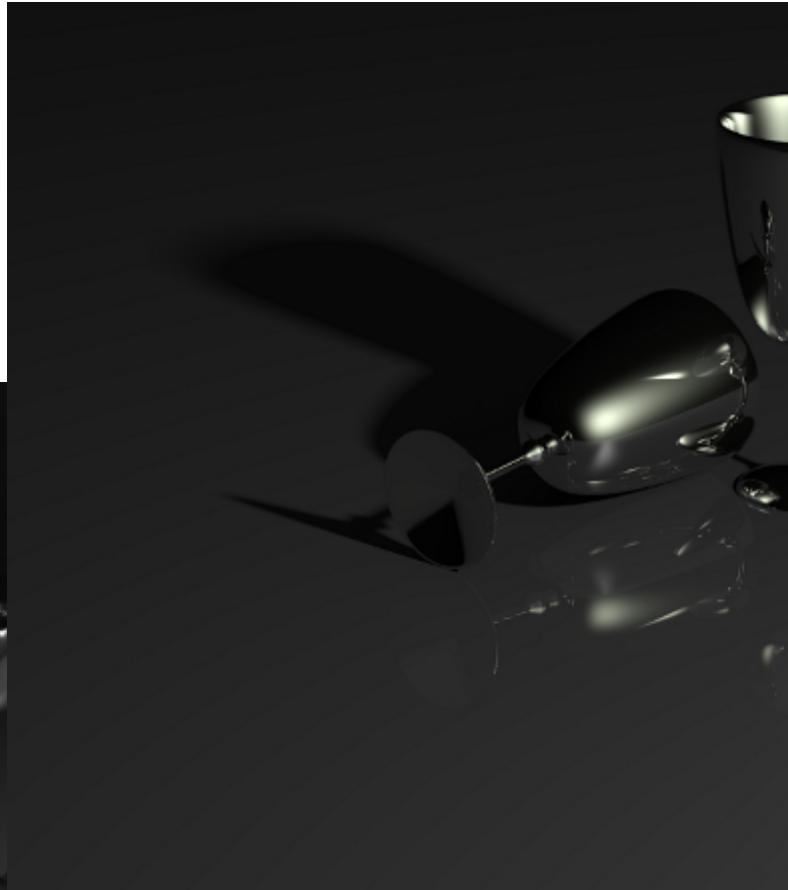
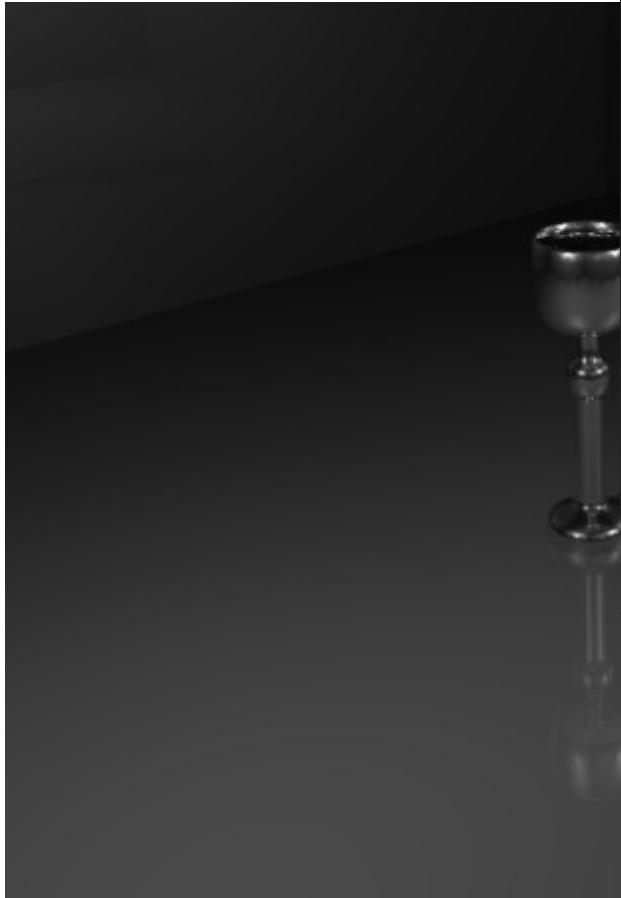
2.34.4 Creating a metallic texture for the goblet:

The metallic look can be achieved by these steps:

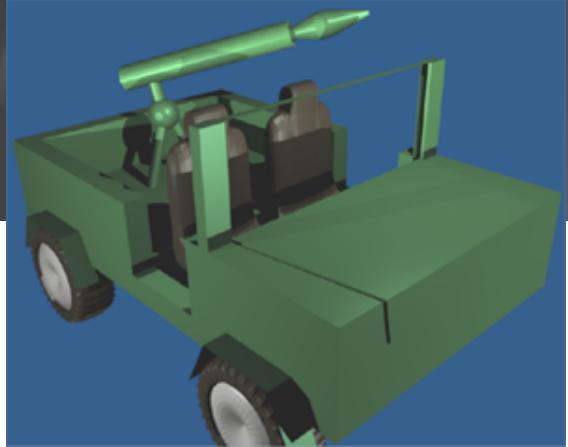
1. **Materials > Diffuse.** Set color Hex value to: **C7C8CB** or you can set it completely black for dark metal. Set intensity to “0.8”.
2. **Materials > Specular.** Set color to white(Hex: **FFFFFF**). Change **Intensity** to “**1.000**”, left of the color-swatch, set the *specular shader* to **CookTorr**. Also change the **Hardness** to value around “**16**”.
3. **Materials > Mirror.** Make sure the Mirror checkbox is checked. Set **Reflectivity** to **1.000** and color to white(Hex: **FFFFFF**). Make sure that under “Gloss” the “Amount” is set to “**1.000**”.

In the lower picture the color is set to black (or close to black). This eliminates the “un-metallic” whitish sheen that the goblet in the upper picture has.

Noob Note: I don't know about those using lower versions but those using v2.5x have a choice of different shaders both for diffuse and specular colours. I strongly suggest that for the same object one should try playing around with the shaders. They can create different effects. for eg. here if you change the diffuse shader type to **oren-nayar** and the specular shader type to **wardiso**, the goblet will have a glossy , finished look.



2.35 Simple Vehicle



Let's make this jeep.

The idea of this tutorial is to learn to face a complex project. A vehicle is a nice object to use to test yourself and find new problems.

First, we must understand that a project does not reproduce the real world; a project shows an idea or thought and will result in a final image or video. Whatever does not appear in the final result is unnecessary to include in the model.

What vehicle should we make? Let's go with the classic jeep. This will allow for a lot of doodads.

Let's decide what objects of the jeep model will need to be made - body, wheels, seats, and a rocket launcher for good measure. Objects we can ignore include the engine, which remains hidden under the hood. There are many additional objects you can make such as a steering wheel to customize your jeep.

2.36 Simple Vehicle: Wheel

There are 2 tutorials for the tires. This and the next tutorial: this is the basic tutorial but the next tutorial is more complicated and you can end up with one of the four different versions.

2.36.1 Techniques

You should already know how to:

- Make a mesh
- Navigate the viewport
- Extrusion
- Create, edit materials

This section will recap and introduce:

- Forming faces
- Subsurfing
- Merging vertices
- Object naming

For our premise, envision jeep tires. They're not too sleek but rather rugged for all kinds of terrain. We need a tire that can handle any obstacle in its way.

During this tutorial we will be primarily using orthographic view. Feel free to switch to perspective view (**NUM5**) from time to time to see how things are developing. You may also want to rotate in the XY plane using scroll **MMB** or **NUM2/NUM4**. Switch back to orthographic view (**NUM5**) to edit.

2.36.2 Model the tire

Hit **NUM1** to set front view (XZ coordinates), then delete the cube.

Create the outside of the tire

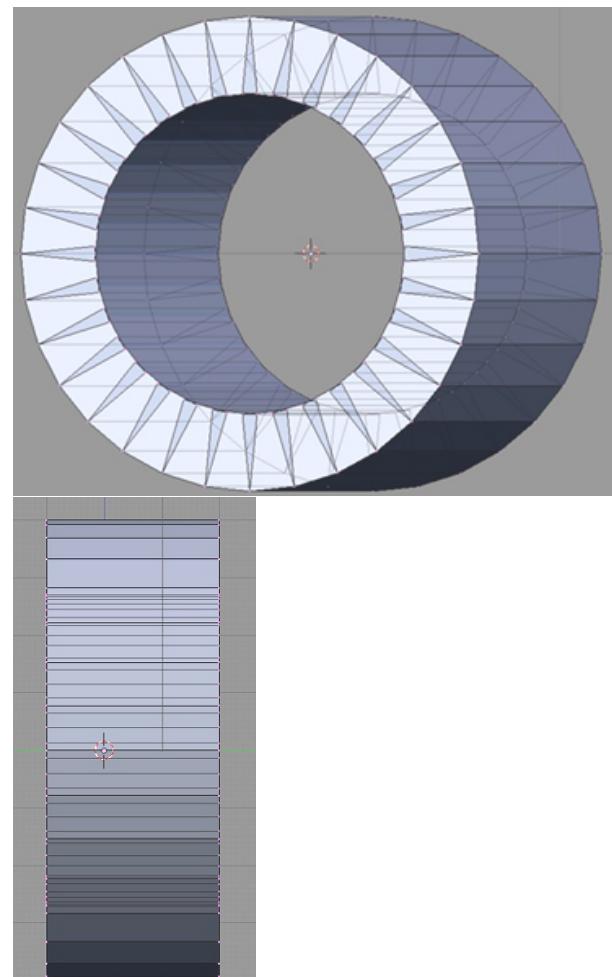
Add a cylinder **Shift+A > Mesh > Cylinder** then in the tool shelf use 32 vertices, set the radius to 4, depth to 3, choose Cap Fill Type: "Nothing", and click on "Align to view". By default, objects are aligned to the *global space* axes. The "align to view" option rotates the cylinder so that it is aligned to the *view space*.

Create the inside of the tire

Switch to orthographic mode (**NUM5** to toggle) and then go into **Edit Mode**.

Select all vertices, hit the **E Key** and directly after that the **ESC key** to make the new faces, then **Alt+EKEY** and choose "Individual faces" and extrude the individual faces into the circle.

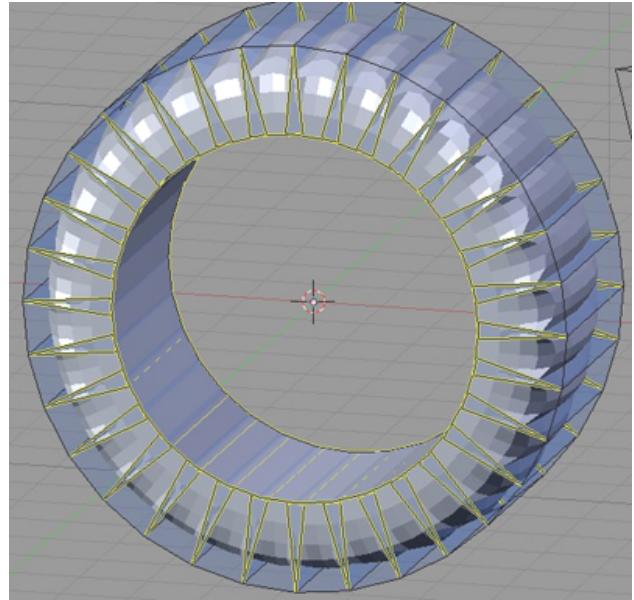
You may either type in -1.2 and hit enter, or hold **SHIFT CTRL** and move the mouse to extrude the faces until the sides come in -1.200 units. Now select all with the **AKEY** and remove doubles by pressing the **WKEY** -> Remove Doubles.



Subsurf the tire

Now it's time to make the tire look like a rugged tire.

Return to Object Mode, and apply a subsurf modifier (use the Modifier menu in the properties header - it looks like a wrench) click on “Add Modifier” and select Subdivision Surface - select VIEW level 1 or 2. The tire will now look like a bead necklace.



Crease the edges

A little creative use of creases will restore our tire.

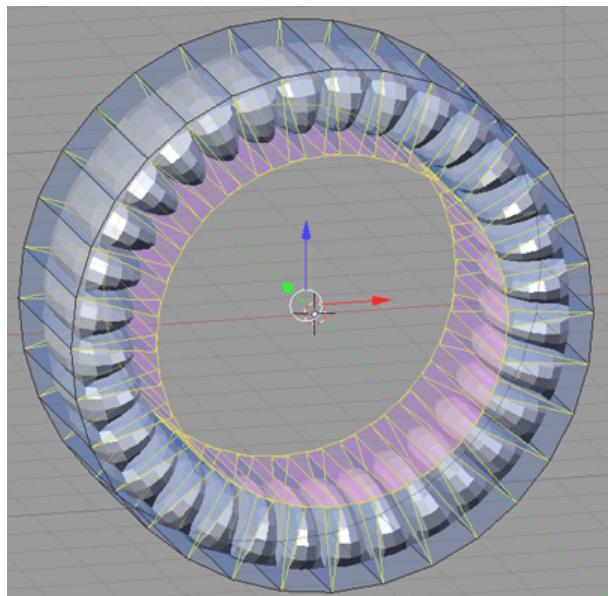
Switch back to orthographic mode (**NUM5** to toggle) if need be then go into to **Edit Mode**. Check that Limit Selection to Visible is off (that is, so you can see the extra edges and vertices).

Enter **Edge Select** mode

Bring up the circle selection tool (**AKEY** to unselect all, then **CKEY**).

Use the scroll wheel to change the circle selection size to be in the center of the tire, between the inside and outside edges. This will select all of the inside edges, as well as the triangles on the side of the tire, as in the picture below. Then hit **Enter**

Now press **SHIFT+EKEY** to Crease these edges - type 1.000, and press **ENTER**, or hold **CTRL** to pull in steps till you see 1.000 in the status bar at the bottom of the view window.



2.36.3 Model the hubcap

The tire is almost done. Let's add a simple hubcap to it.

Create a cylinder

Be sure you're in **Edit Mode**.

Hit **AKEY**, once or twice till all the wheel's vertices are selected.

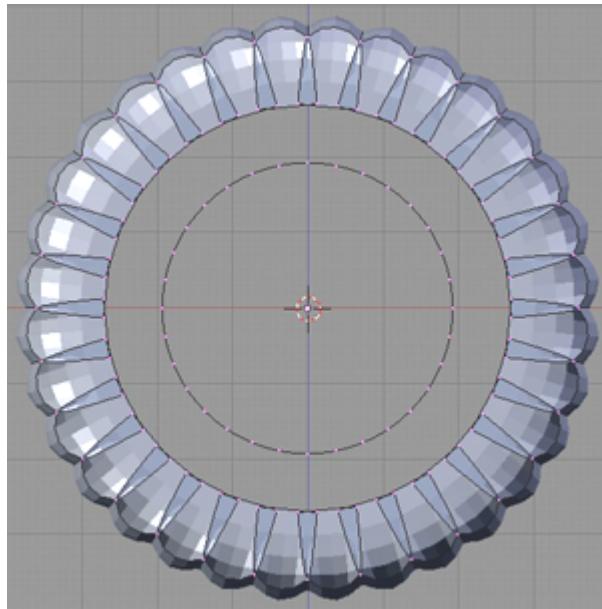
(The scene should be in front orthographic view - hit **NUM1** for front view, **NUM5** for orthographic if not).

Press **SHIFT+SKEY**, from the popup menu, choose “Cursor to Selected”, to put the cursor at the center point of the existing tire.

Hit Shift+A => cylinder with 32 vertices, radius of 1.9, depth of .5, choose Cap Fill Type: “Nothing”, and click on the checkbox of “align to view”.

Turn orthographic view off: **NUM5**. Hit **NUM7** for top view.

Hit **GKEY**, then **YKEY**, then type 2.2, and hit **ENTER** to move the hubcap into part of its eventual location and a place we can work on it.



Create the outside of the cap

In top view, hit **AKEY** once, so that nothing is selected.

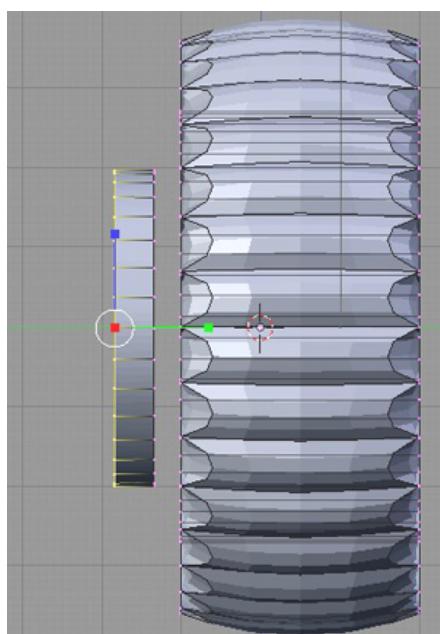
Zoom in with the **MMB** till the hub cap fills most or all of the view.

Switch to **Vertex select** mode. Make sure “Limit selection to Visible” is off (that is, so you can see the extra edges and vertices).

Hit **BKEY** for box select, then holding the **LMB**, drag the box to enclose the vertices along the top edge of the hubcap.

Hit **SKEY**, then **SHIFT+YKEY** to only move in the XZ axis, then type in $.35$, and hit **ENTER**

Hit **GKEY**, then **YKEY** to only move the Y axis, then type in $.35$, and hit **ENTER**



Create the Axle Cover

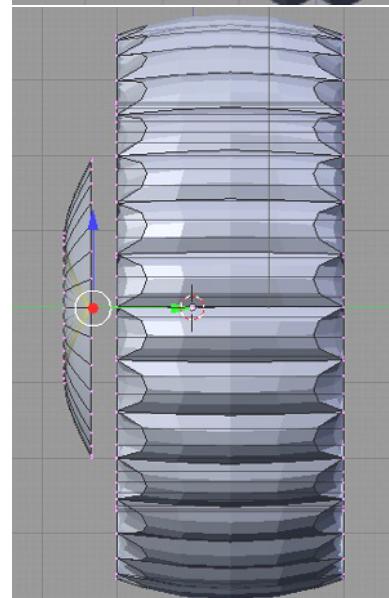
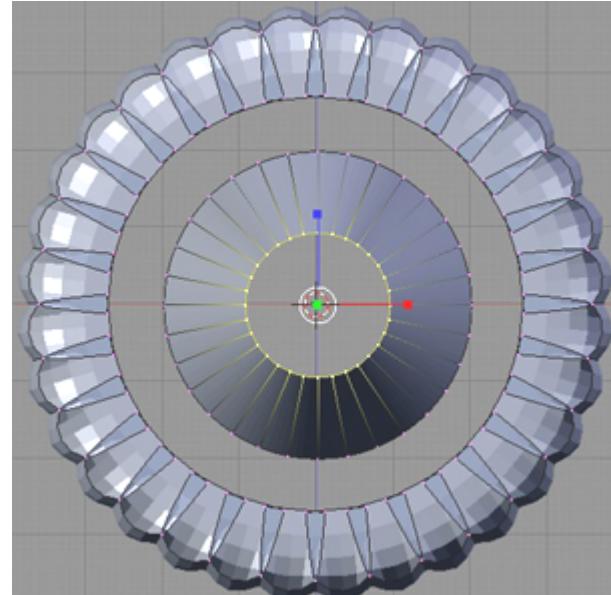
We'll merge these vertices together to create a flat surface.

Hit **Alt+EKEY**, on the popup select **Edges only**, then hit **ESC**, to create the edges we will need.

Hit **ALT+MKEY** on the popup pick **At center**. Blender will reduce the 32 vertices to 1.

Hit **NUM3** for side view.

Hit **GKEY**, then **YKEY**, to only move the Y axis, then type in -0.4 and hit **ENTER**.



Final sizing of hub cap to tire

The final mesh editing is to scale the hub cap to a size that is slightly larger than the hole of the tire.

Hit **NUM7** for top view, **NUM5** for orthographic if needed.

Position the mouse over the hubcap, and press the **LKEY** to select the entire hubcap.

Hit **SKEY**, then **SHIFT+YKEY** to move only the XZ axis, then type 1.48, and hit **ENTER**

Hit **NUM3** for side view.

Hit **GKEY**, then **YKEY**, to move only the Y axis, then type -1.11 (use -0.77 if you want your hubs sticking out) and hit **ENTER**.

Renaming the Wheel

The last thing to do is to rename the wheel so we can find it easier later.

Enter Object mode and select the wheel only.

In the outliner window you'll see the tire called "Cylinder". This name was created because we started with a cylinder mesh.

Click on the name with the right mouse button and click on Rename - rename the object to something like 'wheel'. Save your file where you'll find it later and continue to the next step.

2.36.4 Extra



Rendered tire

Change the materials to make it look like a tire. As you have seen in previous tutorials, one object can have multiple colors/textures.

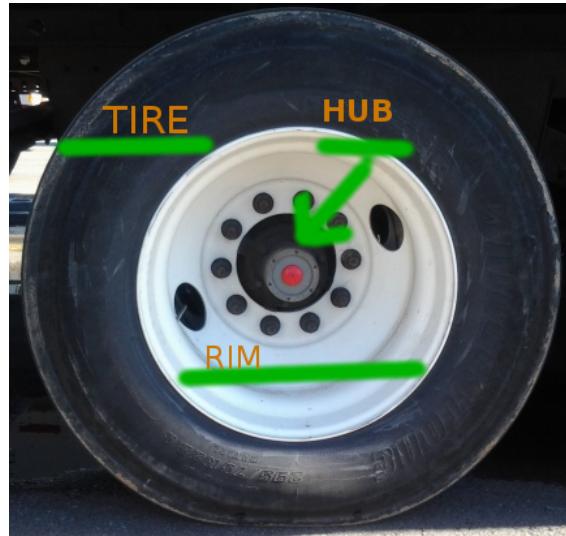
If you'd like to review how to do this, then refer to the materials section for an explanation on how to, or to the Blender manual: Multiple Materials

2.37 Simple Vehicle: MudTires

There are 2 tutorials for the tires. This and the previous tutorial. The first is basic but this is a more complicated tutorial with a total of 4 different versions.

Congratulations on making it this far; you've proven you have what it takes to finish this book. At this point you don't need me explaining the simple things like the differences between or how to select vertices, edges or faces. Feel free to swivel the camera this way and that. I'll leave it to you to decide when to turn snap on or off "Limit Selection to visible". At every step I hope you think, "Ah-ha, now it looks more like a tire!" since that's essentially what I did. I'm writing this tutorial because I don't want this book to become outdated. Also... I really like tires!

2.37.1 Build the Tire



A real world wheel

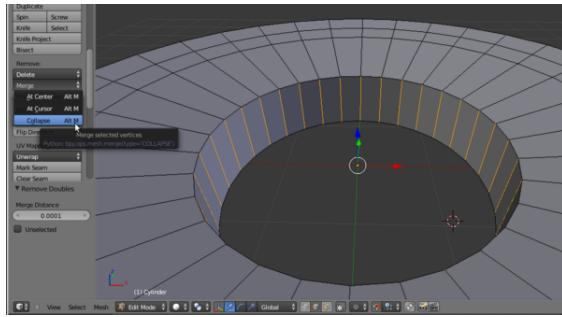
This is the real world wheel we'll be modelling. It has three parts: 1) tire 2) rim 3) hub. We'll create the wheel laying down starting with the tire and working our way in.

The wheel

Delete the default cube and hit Num7 to go to the top view.

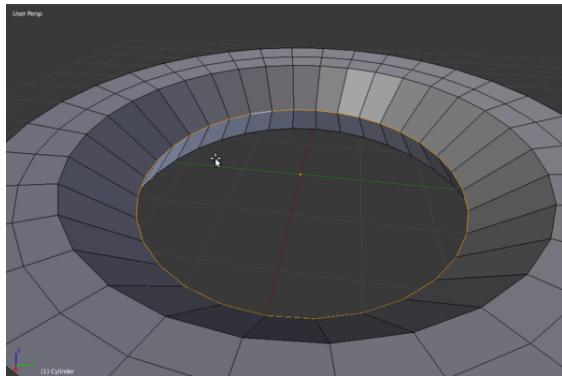
Center the cursor and add cylinder with 32 vertices, radius 4, and depth 2. Set Cap Fill Type to nothing. Set snapping to Increment: Shift + Ctrl + Tab and enable snapping: Shift + Tab .

Next we will make a series of cylinders which will become the tire and the rim. Switch to edit mode. Extrude the cylinder inwards by pressing E Esc S Shift + Z 0 . 9 Enter . Do this a total of two times, then extrude another cylinder inwards by 0.8.



only select the vertical edges

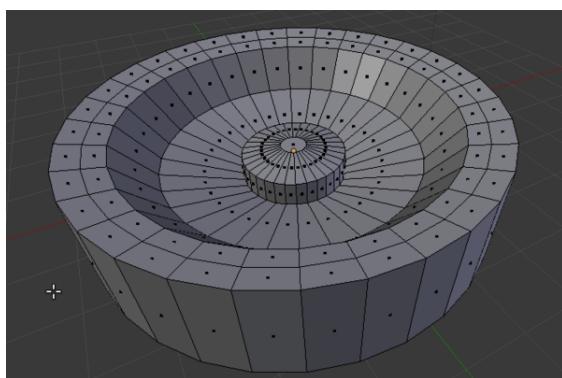
Unselect everything (A) then select only the vertical edges of the innermost cylinder, as follows: ensure “limit selection to visible” is turned off, switch to vertex-select mode, circle-select the innermost vertices, switch to side view (NUM1) and edge-select mode, block-unselect the top edges with Shift + LMB , and do the same for the bottom edges.



merge collapse the edges

Merge-collapse (Alt + M >> Collapse) these vertical edges to create a circle that's vertically in the center of the other cylinders.

The Hub



completed simple wheel

The inner ring should be already selected. Extrude it and

scale to 0.4. Move the new circle down by 0.5, then extrude it up (E Z) by 0.5. Extrude and scale to 0.25. Move the circle up by 0.15, and lastly press F to make a face in the inner circle.

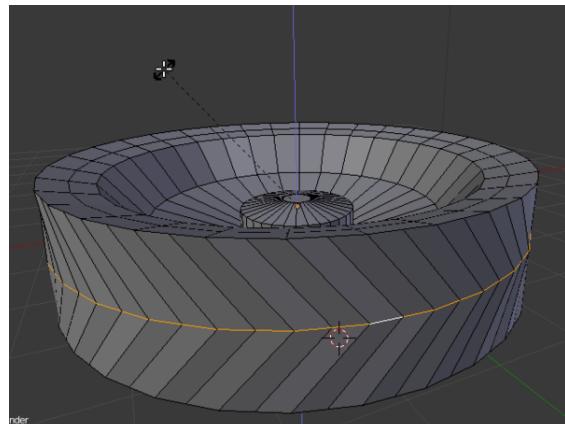
To be safe before saving highlight all and remove doubles. We can add different types of tread to this simple tire. If you're feeling ambitious come up with your own tread design. Save your work now.

2.37.2 Finish your tire several versions

Working on the vertical faces of the outermost cylinder in face select mode.

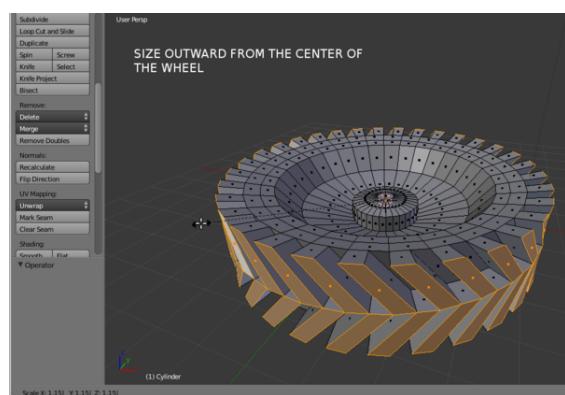
Type 1

Subdivide the outer faces one time, and rotate, with Z-Lock, the newly created central edge by 12.



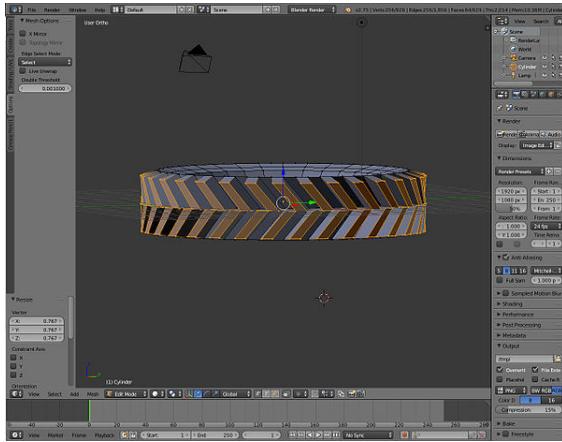
rotate the middle edge by 12

Stagger select every other outer face and extrude size out 1.15 (“Ekey” then “Enter” then S and then 1.15). Make sure your sizing outward from the center of the wheel. For quick selecting, Alt-select the two rings of faces and press Select-Checker Deselect



select tread pattern and extrude outward by 1.15

What you can also do is Alt+E click on individual faces, 1.15 and then “S key” scale it down some points.

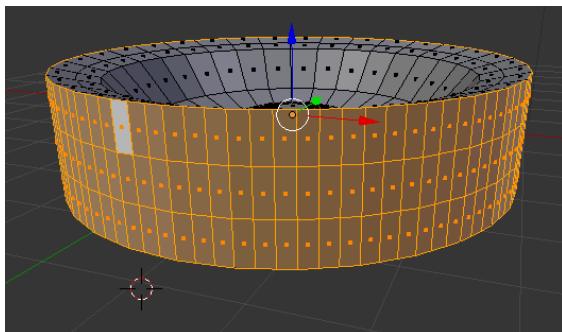


nice round wheel

Type 2

This tread is a little trickier but the extra work is worth it.

Subdivide the outer faces and change the number of cuts to 2. There are now 96x3 outer faces. Think of these as being on 3 levels bottom middle and top.



creating faces

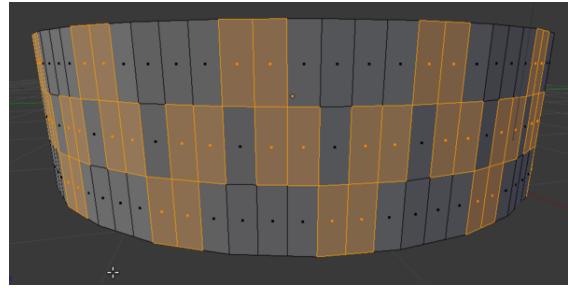
Select 2 of the bottom faces skip 4 faces then select 2 more go all the way around the circle. The pattern for the middle level is select 2 skip one select 2 starting at the top left vertex of the bottom level. The pattern of the top is the same as the bottom select 2 skip 4 start at the top right vertex of the middle level just not the same middle selection that the bottom is... see the picture :)

Extrude size out 1.15 from the center of the wheel “Ekey” then “Enter” then S then 1.15)

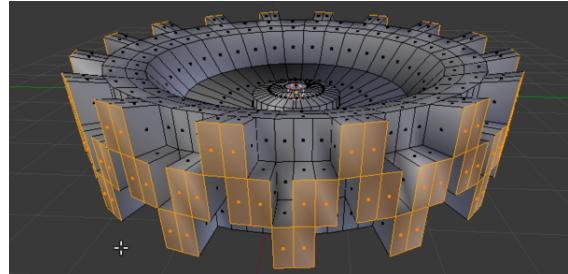
Type 3

Evenly loop-cut the faces of the outer cylinder

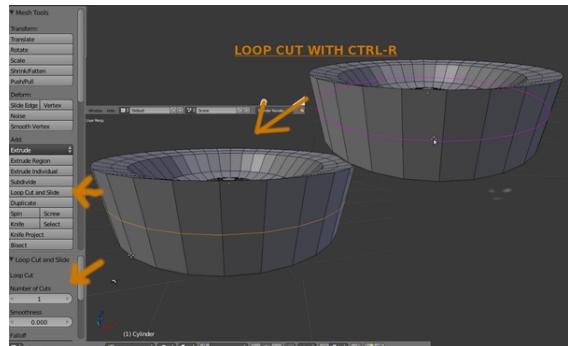
CTRL-R, center the mouse on the outer cylinder -> Enter -> Enter. Now subdivide the outer cylinder. There are



select tread pattern

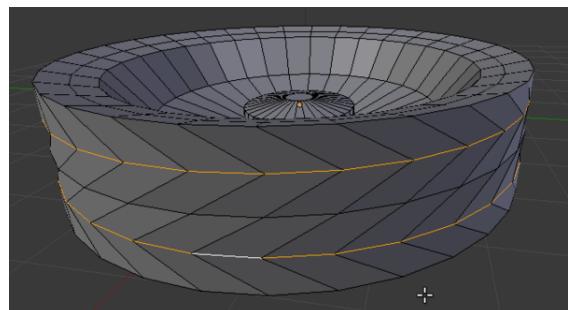


extrude outward to create tread



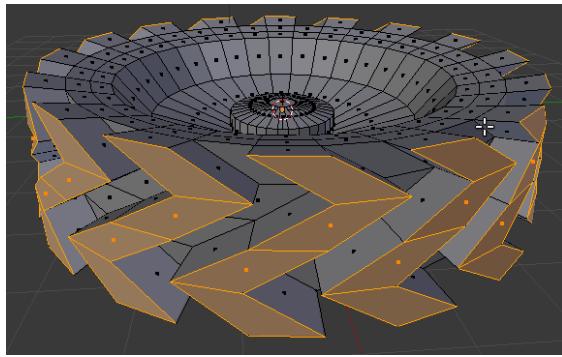
loopcut center of wheel

now 3 inner edges. rotate, with Z-Lock the top inner edge by 12 and the bottom inner edge by -12.

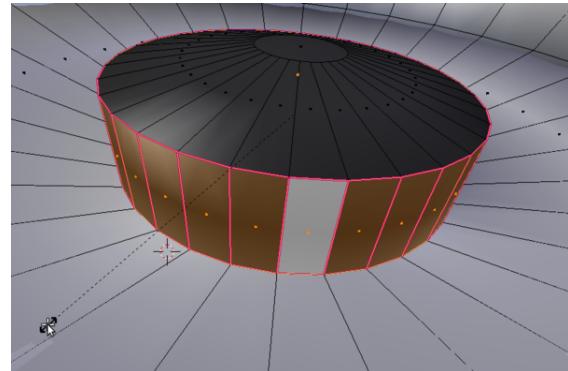


rotate edges

This time make a Z-pattern with the faces and extrude size out by 1.15 from the center of the wheel “Ekey” then “Enter” then S then 1.15).



select tread pattern and extrude outward

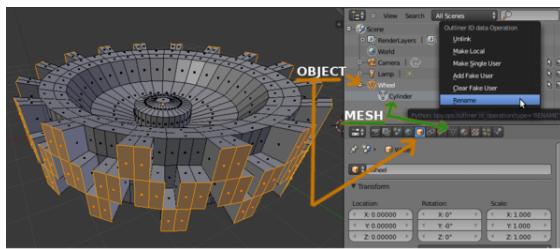


crease the hub

2.37.3 Finishing

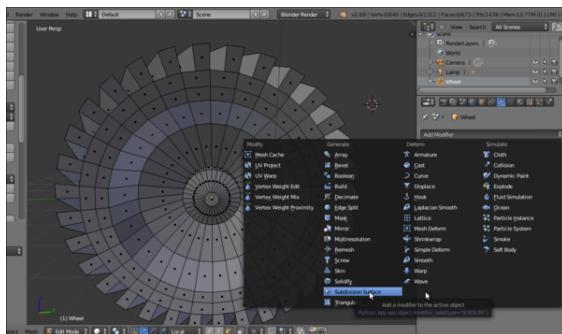
To get the wheel standing up lock the y axis and rotate everything 90°

The wheel object as well as the wheel mesh are both named Cylinder in the outliner lets rename both Wheel. Right click the name and select rename



rename object/mesh wheel

Lets add a sub modifier and set shading to smooth.



subdivide to smooth things out

Crease the hub edges. Shift EKEY 1 enter. By the way the easiest way to remove a crease is just to add a negative crease. Shift EKEY –1.

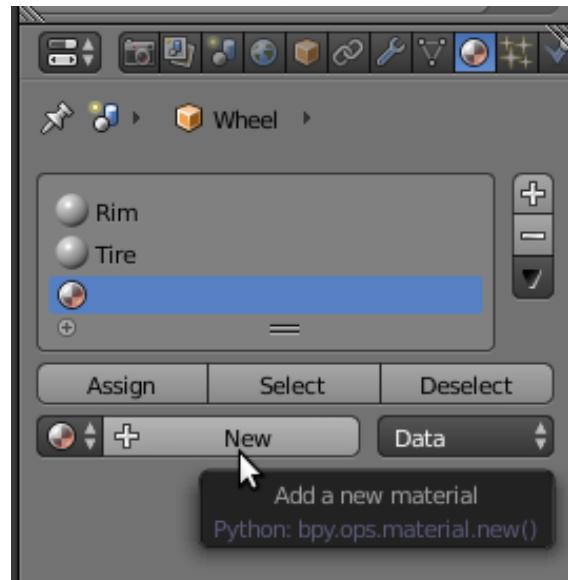
2.37.4 Colouring

Now for some color. By the way, think of diffuse as being the base color and specular as being the reflected color. If

you're curious about words like lambert and fresnel just look them up in your favorite dictionary.

In the properties menu select the materials icon.

Create three material slots by clicking the plus button that's above the minus button 3 times.



after you click new rename the material

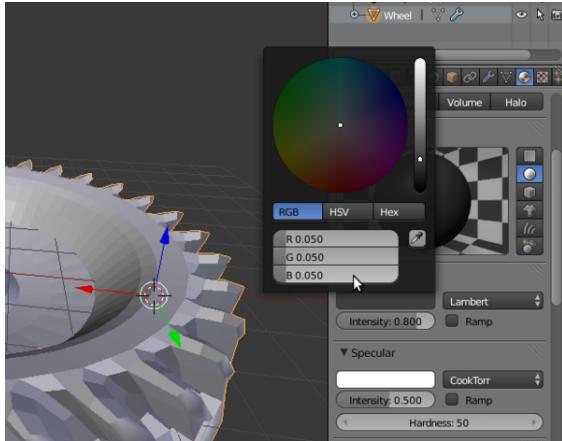
Select each material slot and click new to add a material to it. Name the first material Rim, the second Tire and the last Hub.

We'll leave the rim default white. Select Tire and change the diffuse color to almost black RGB of 0.010 and the specular intensity to 1. This gives our tire a very shiny black look. Give the hub a dull gray look by setting the diffuse color to RGB 0.05.

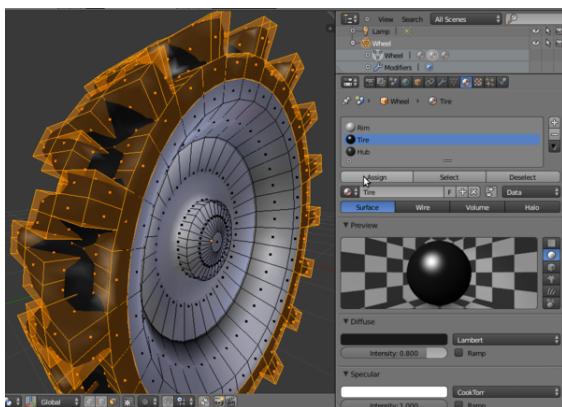
The first material added becomes the default material for the entire object. So we only need to select the parts of the wheel mesh that are not rim.

Select the tire that's the two outer cylinders- with the tire material highlighted click assign.

To make selecting the hub easier close the subsurf eye in

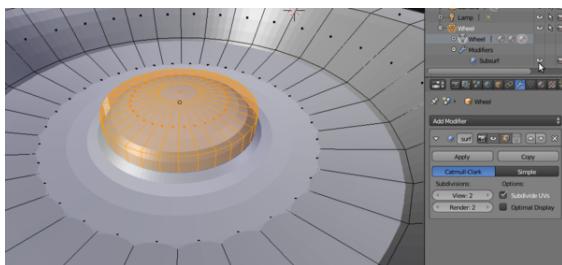


create dark gray by lowering all colors to 0.05



assign the tire color to the tire mesh

either the properties or outliner menu. Assign the hub material to the hub mesh. I've left adding the red hub oil seal up to you.



turning subsurf off makes selecting easier

That's it! Hopefully my instructions weren't too painful. Be sure to save and good luck!

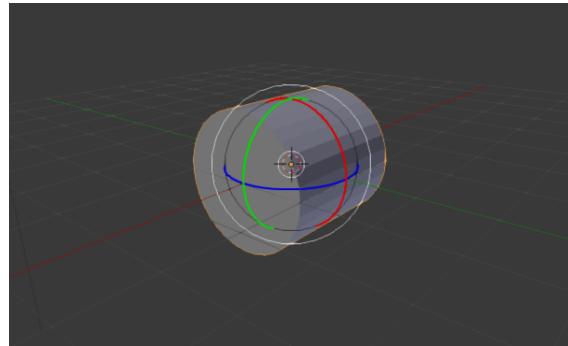
If you want you can create a more complex tire. Otherwise you can skip this.

2.37.5 Creating a more complex tire

Start by loading a basic cylinder and rotating it so it rests on its side.

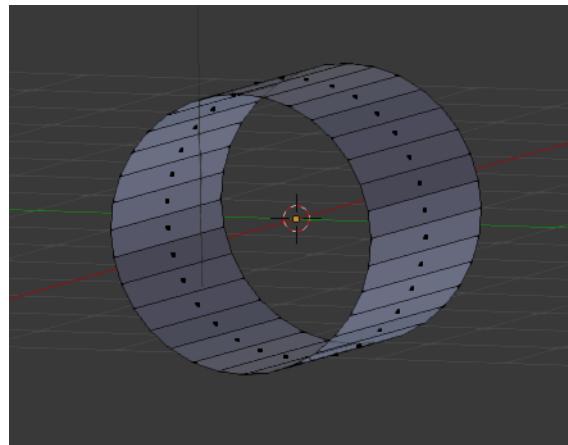


The final Tire rendering



Cylinder

Next, remove the faces from the top, and bottom of the cylinder.



No Faces

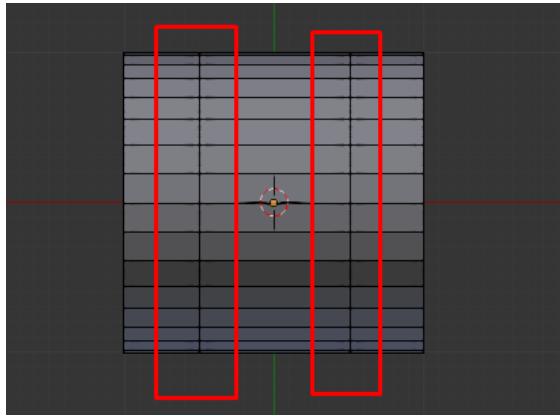
Now Press the num5 key to enter orthographic view.

Then press num7 to view the front of the cylinder.

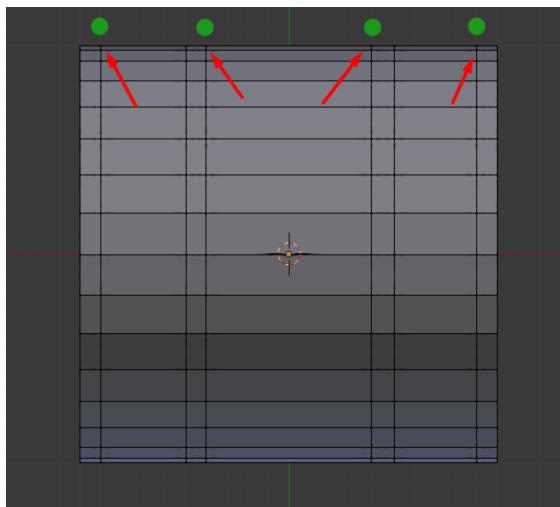
Then click on "Loop cut and Slide" in the tools shelf and make a cut of ".5" to each side as shown in the picture (ctrl+R -> 3 -> Enter -> Enter) select the middle then (X -> edge loops).

Next, make 2 more cuts .4 on each side (ctrl+R -> 4 -> Enter -> Enter) select the unneeded then (X -> edge loops).

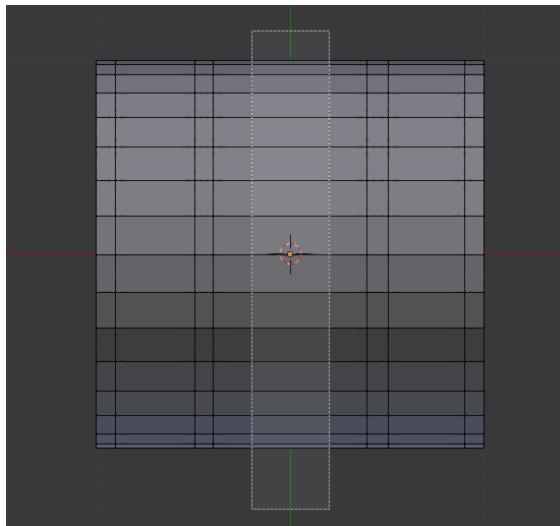
Next, press the b key to enter box select mode.



2 Cuts



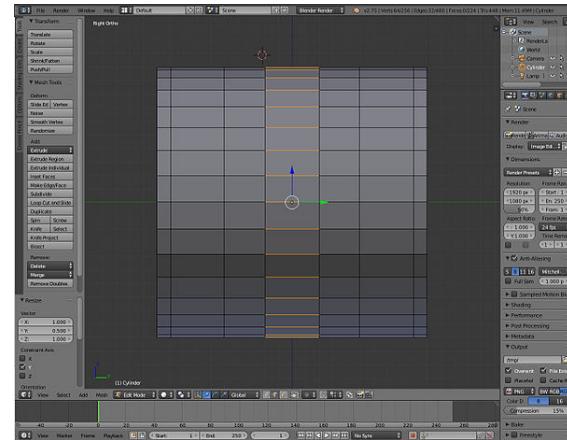
Total Count Of Cuts



Boxselect

you will want to select all the middle faces.

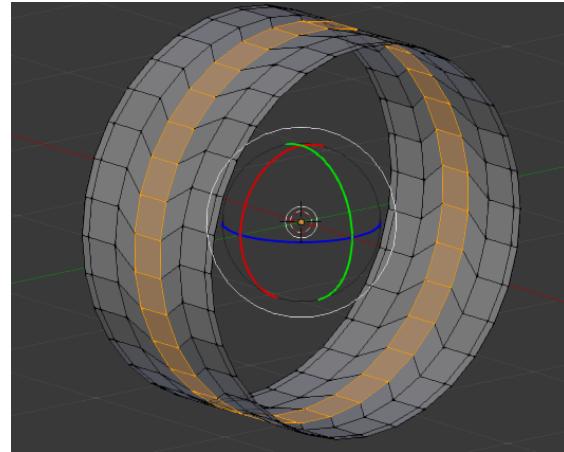
Next. type "S key" then X or Y, dependent on direction your cylinder is in, then .5



Scale The Middle

Then press the A key to select everything, then "S key" -> X or Y, and type .5 to squish the whole wheel.

Then use the "B key" to reselect the middle faces. Now, with the Middle faces still selected rotate them so they look similar to the picture shown by typing "R key" "X key" or "Y key" and then "10".



Rotate The Middle

Next, select all the faces beside the middle and in the middle and rotate them all in the opposite direction so it looks similar to the picture shown by typing "R key" "X key" or "Y key" and then "-10".

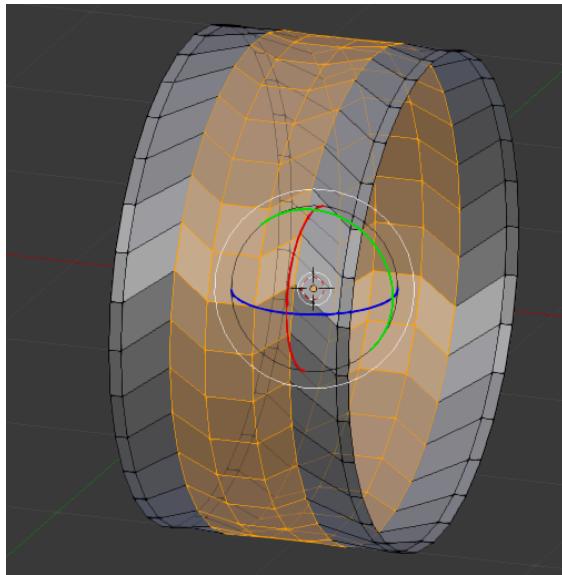
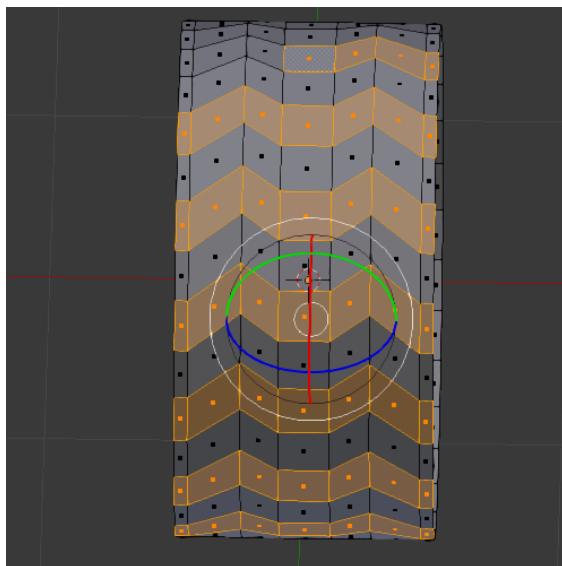
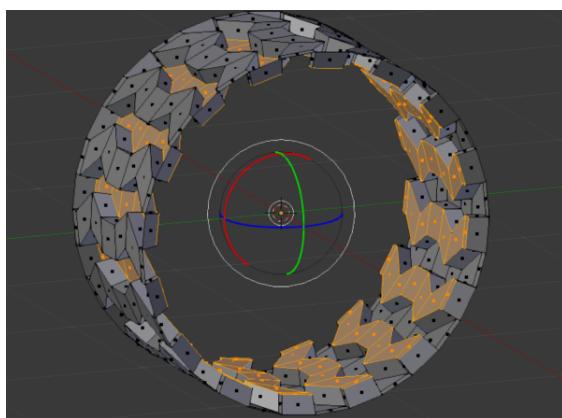
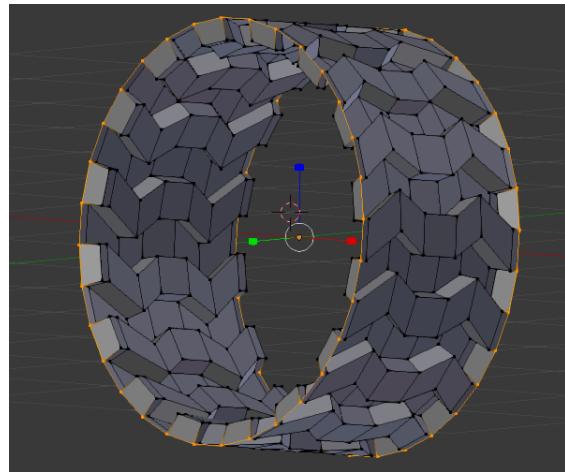
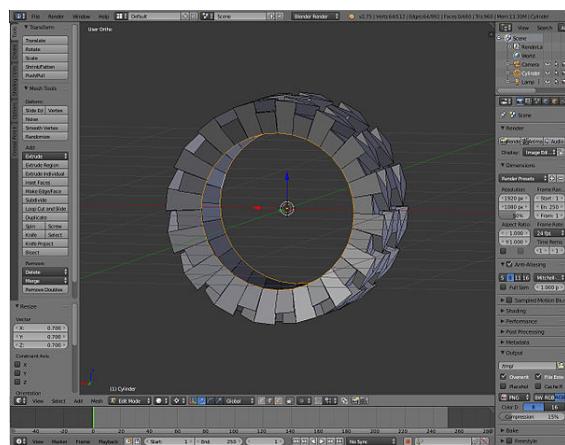
Next press the A key to unselect everything.

Then start selecting faces as shown in the picture. Continue this all the way around the wheel leaving 1 space of unselected faces in-between each row.

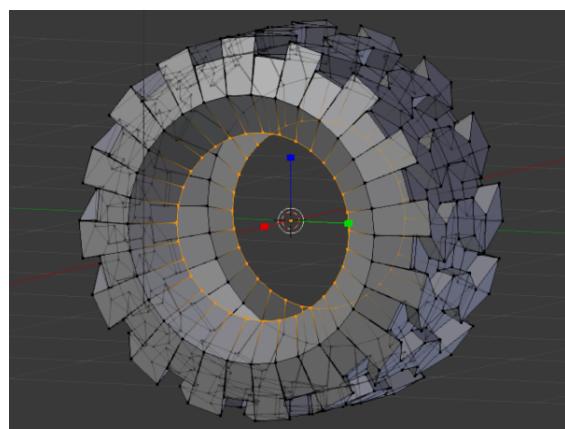
With the faces selected press "Shift+S" then "Cursor to Center" then EKEY then ESC and then "S key" to scale them. While scaling type ".9" then press enter.

Next, select all of the side edges as shown and Then Press the "S key" and type .7 then press enter.

Type "S key" then X or Y, and then type "1.5" to move both sides outward slightly.

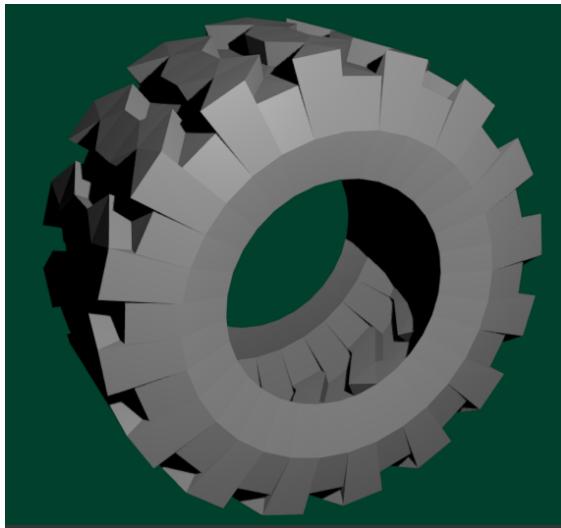
*Rotate The Middle And Sides**Select The Faces**Extrude And Scale**Outside Edges**Sides The Tire*

Type “E key” then “Enter”, then the “S key” type “0.75” as shown in the image.

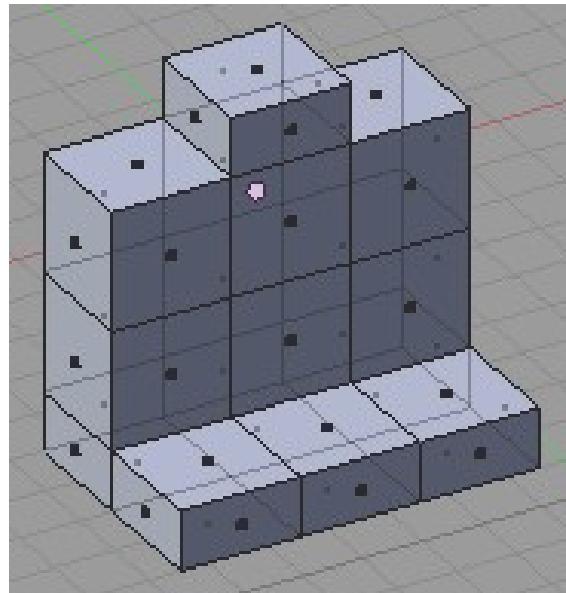
*The Inner Of The Tire*

You're done!

Feel free to add the hub cap from the previous wheel, or one of your own design.



The Render Of The tire



Basic setup of shapes.

2.38 Simple Vehicle: Seat

2.38.1 Techniques

You should already know how to:

- Make a mesh
- Navigate the viewport
- Extrusion
- Subsurf
- Crease edges

This section will recap and introduce:

- Loop Cut and Slide (Loop Subdivide)
- Small, consistent vertex movement

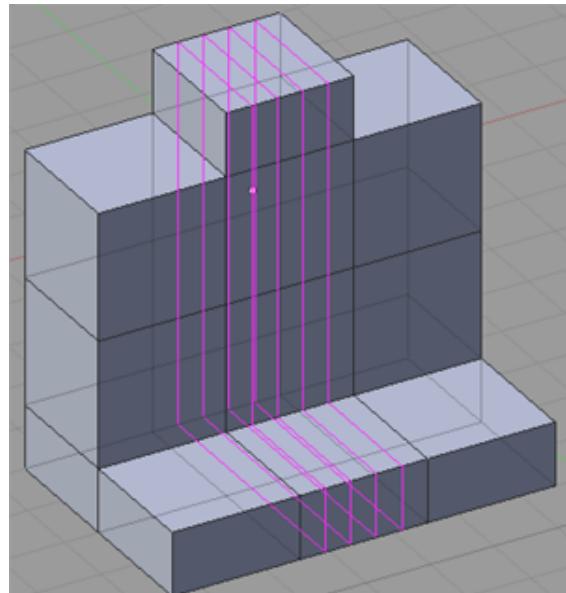
The design will be an all-terrain bucket-type seat.

2.38.2 Extrude the Seat

Start in NUM1 view of the default cube and rename it.

Extrude the cube multiple times to make your basic shape. In this example a 3x3x1 block composes the body with one cube coming out the top for the headrest, and the bottom cube's front faces extruded out to create the seat.

Noob Note: best practice is use “Face” selection mode, select the face, or faces you want to extrude with the RMB , then hit E , select an axis to move on by pressing the X , Y , or Z keys, and pull with CTRL held down.



Multiple Loop Cut and Slide.

2.38.3 Add cushion seams

Noob Note: Before starting on this, you'll need to make sure you don't have any unnecessary faces **inside** the seat, or else you'll get strange cushion seams.

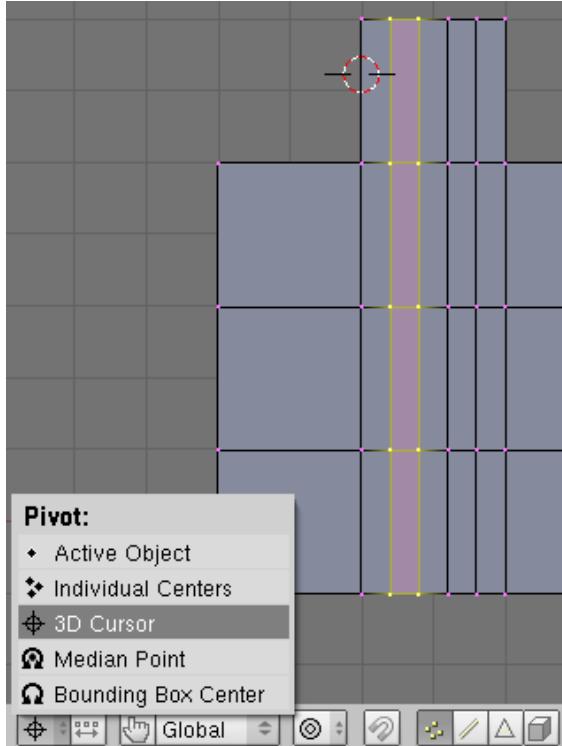
To add a little texture to the mesh, we'll add some cushion seams. Use Loop Cut and Slide (CTRL + R and you'll see the pink selection loop. You can use the NUM+ key to increase the number of loops made at the same time. Use mouse wheel or press NUM+ 3 times to form 4 loops and LMB the center column of blocks. You may also find it easier to add them one at a time in the correct place, than inserting them and then moving them.

(To get multiple Loops instead of pressing NUM+ just

press the number of loops you want in this case NUM4 , this is a fast easy way to achieve this.) You can use a mouse-wheel as well. Finally pres “Enter” 2 times.

Noob note: If the loop comes up with green lines rather than purple you have gone one step too far, just press ESC and try again. When you see the purple lines use your MWH or the NUM+ button.

2.38.4 Position the cushion seams



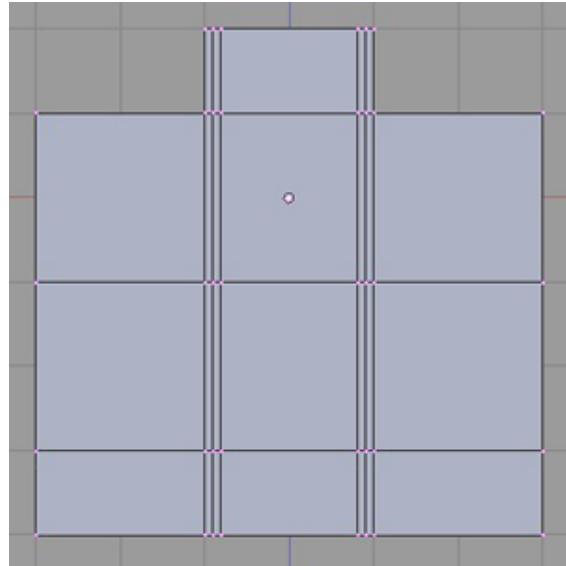
Where the 3D cursor should be placed and what to select

Go into ZX view (NUM1) and make sure the view is orthographic (NUM5 to activate/deactivate orthographic view). Place the 3D cursor on one of the sides of the head rest (LMB , or SHIFT + S to move cursor).

The idea is to move the 4 loop cuts just created away from center to the sides. Set the pivot to 3D cursor and select the two closest loop cuts. Scale (S) it down on the X axis (X) to 0.3. The goal will be to have the loop closer to the cursor to go into the cushion to become a seam. Now, do the same thing for the other side of the head rest.

Noob note: The easiest way to select the two closest loop cuts is to first select (RMB , SHIFT + RMB) one edge of each loop, and then go to Select Edge Loop (SPACE -> Select -> Edge Loop). Or hold down ALT when selecting one of the loop’s edges. This should select the entire loop. In order to select more than one loop, hold down SHIFT as well.

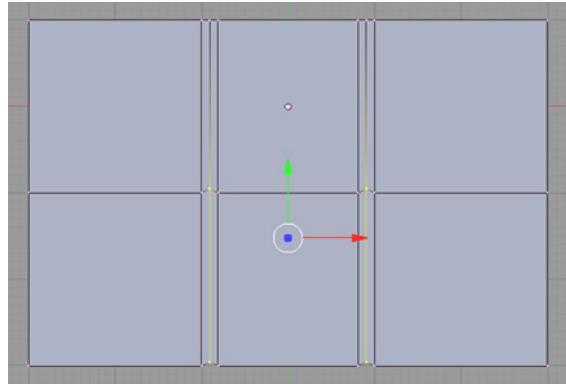
Noob2 note: in Ortho view with ‘Limit selection to visible’ disabled, just box-selected with one drag of the



Positioning the seams.

mouse.

2.38.5 Add Depth to the seams



Vertical seam creation.

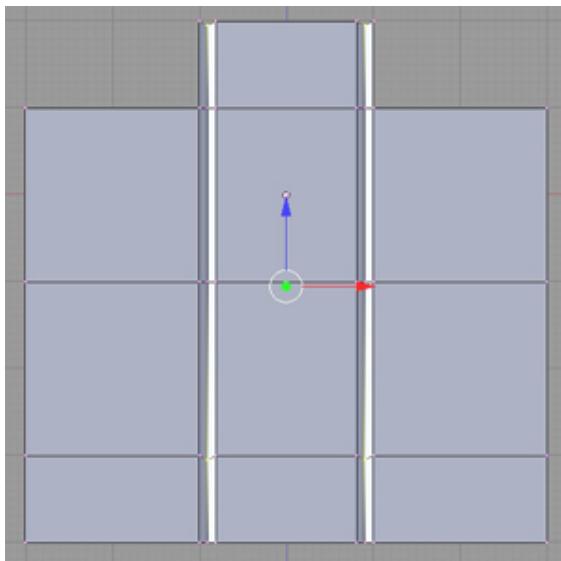
In overhead view (NUM7), select the vertices in vertical parts of the two front facing seams of the seat back, grab them (G), move only along the Y axis (Y), and type in **.05 ENTER** .

Switch to NUM3 view and move the vertices in the horizontal parts of the same two seams, grab them and move them down by moving them –0.05 along the Z-axis.

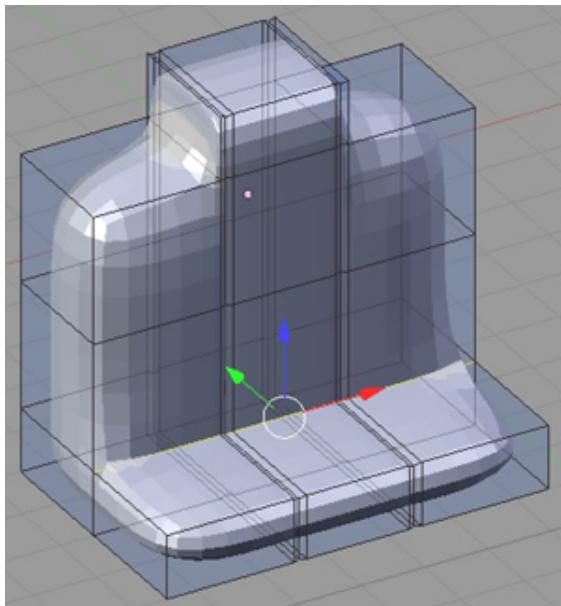
Noob Note: I’ve found that an easier way to do this is selecting each loop of the seam one by one and scaling to .95 (make sure the pivot is set to “Bounding Box Center”)

2.38.6 Subsurf the seat

Add a “Subdivision Surface” Modifier to the object. Hit in the Tool shelf: “Smooth” under “Shading” **Not Nec-**



Horizontal seam creation.



Subsurf seat

essary: Select the edges between the back and seat and crease ('SHIFT + E) them. Crease any edges you feel like to create your perfect jeep seat. **Noob note:** it's best to be in **Edge** select mode when creasing

A subsurf level of 2 or 3 looks best, and don't forget to change the render level to 3 or 4.

Select all (A) then either hit the "Set Smooth" button at the bottom of links and materials, or hit W and select "Set Smooth" for a much smoother subsurf.

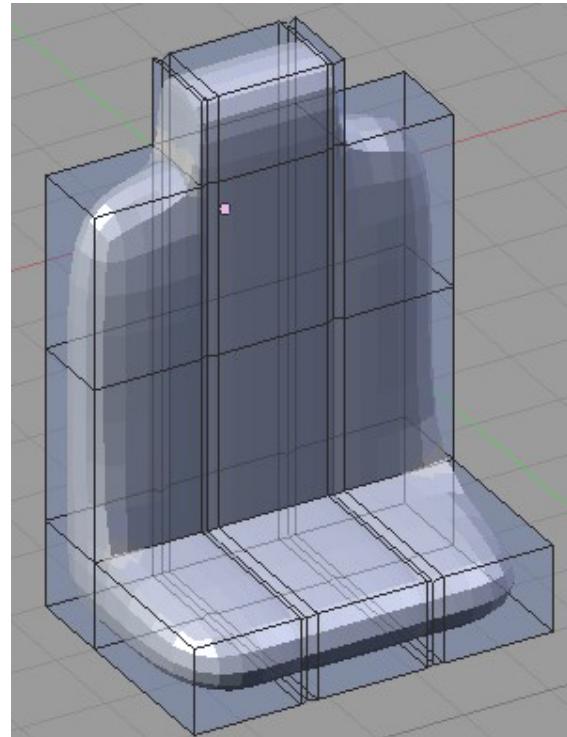
Noob note: If your seat is noticeably misshapen after adding the subsurf modifier, you may just have to delete internal faces in your model. TAB into edit mode, and hit Z to get into wireframe mode. Click the face select button and look for faces that are totally inside the model. There

will probably be a couple vertical faces (in YZ-plane) under the seams in the seat. I found a few elsewhere, also. Deleting all these cleared everything up.

Noob note 2: Another way to remove the extra faces (which can cause the "seams" to be very deep) is to go into edit mode, select your whole seat (A), then hit (W) and select "Remove doubles" from the menu. This is much quicker than finding them individually, and should solve the problem.

Noob note 3: The problem may also be solved by going into edit mode, using A to select all, and using SPACE > Edit > Normals > Recalculate Outside (CTRL + N).

2.38.7 Resize the seat



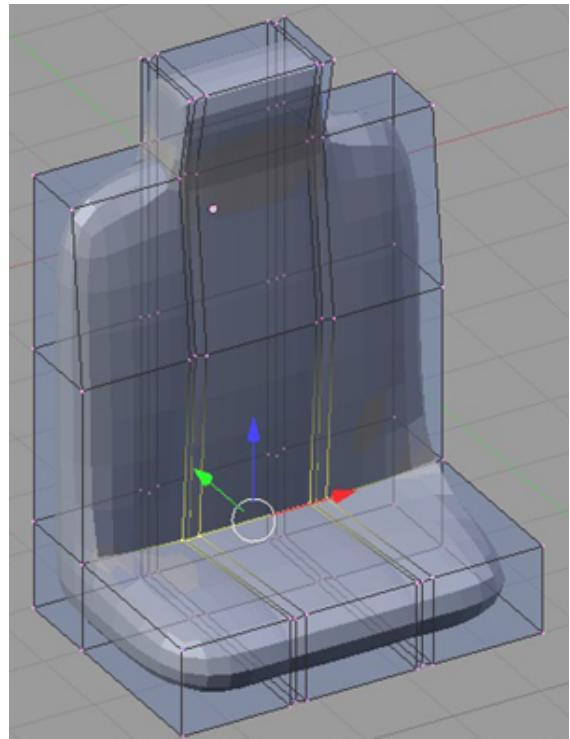
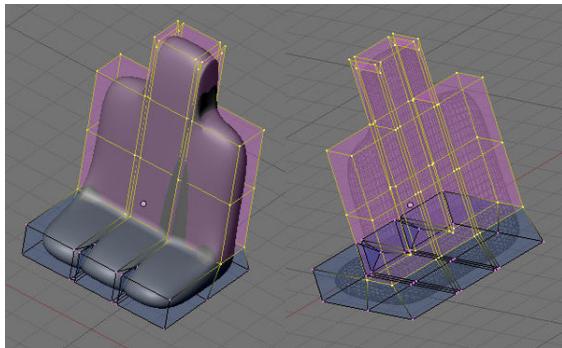
Change the widths.

Next, resize the seat's height and width.

Note: Be sure to change the rotation/scaling pivot back to center point!

To make the whole seat narrower in width, select all A twice, then hit S , followed by X then type 0.8 and press ENTER .

To make the seat back a bit narrower in thickness, select around the seat back (in vertex mode) with the circle select C . Once you have it completely selected on all of the sides, hit S , lock axis with Y , and type 0.75, and hit ENTER .



2.38.8 Final touches

This final seat renders to:



Rendered seat.

Making it concave.

More concave

Also the seat can also be made slightly more concave to look like it would hold a person better.

Leathery look

To give the seams a leathery cord look, Hit A twice to select all, then W and choose Subdivide Fractal on the popup menu (in 2.6x select Subdivide and then press F6 and choose 1 for fractals); just keep the defaults and the seams will look like a bunch of vines until you render it and they look like leather seams, and set in “Material” in the “Properties header” diffuse color to deep black.

2.39 Simple Vehicle: Rocket Launcher

2.39.1 Techniques

You should already know how to do:

- Previous Simple Vehicle techniques

This section will recap and introduce:

- UVspheres
- Changing object's center

2.39.2 Overview

Two assumptions are going to be made here. One is that the rocket will not be launched in the future (use separate objects if you want to do that). The other is this is going to be a simple, simple design.

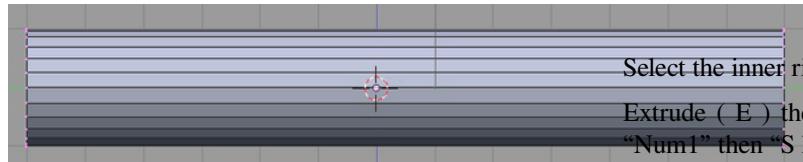
If you want to add options to your gun (think sight, trigger), go for it!

2.39.3 Create the Launcher

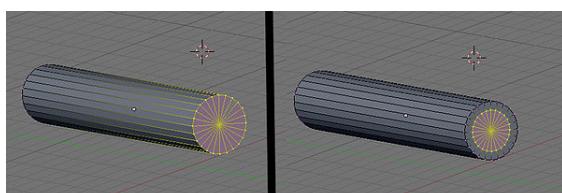
Start a new file and delete the default cube.

Add a cylinder

In NUM1 view, add a cylinder mesh with 24 vertices with “Cap Fill Type” at “Triangle Fan” and set it to “align to view”. We’ll use 24 because the default 32 is overkill and will only increase rendering time. Rename and elongate the cylinder along the Y axis (“S key” then “Y key” then “6key”). This will be the length of the launcher (minus the rocket).



Hollow the cylinder



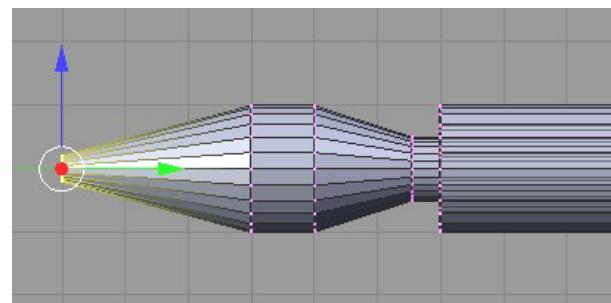
Circle Select (C) the vertices at one side, then extrude (E) them, and press ESC to create a copy of the vertices. Scale (S) them by 0.7.

Do the same at the other side.

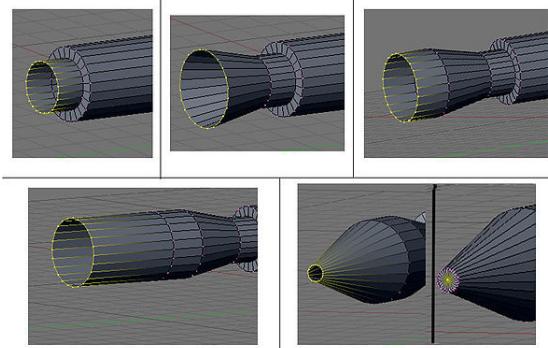
Extrude at one side the inner circle vertices (E) then “Enter”, then press “N key” and change the value at “Z” in 1.2 or -1.2 dependent on what side you take. Then press hit “A key” till you have the whole cylinder selected then press “W key” and say “Remove doubles”.

2.39.4 Create the rocket

For the purposes of this tutorial we will add the rocket on the left end of the launcher.



For a one piece rocket + launcher



Select the inner ring of vertices using circle select (C)

Extrude (E) the “edge” along the Y axis (Y) press “Num1” then “S key” then “0.95” select the all the faces second from the outside and press “X” then “Only Faces”. Select the inner ring vertices then “E key” “Y key” then “0.5”.

Extrude the “edge” along the Y axis with 1.5, and scale the new edge by 1.5.

Extrude the “edge” along the Y axis with 1.

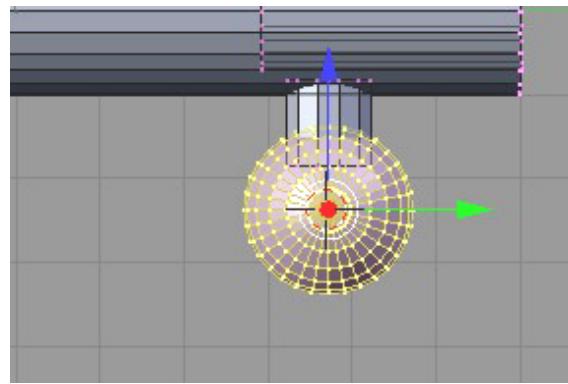
Extrude the “edge” along the Y axis with 3, and scale the new edge by 0.2.

Extrude the “edge”, hit ESC , and merge (ALT + M) at the center to form the face for the nose

For fun you could extrude the left end of the rocket with -11 in the cylinder and then “F key”. you can separate the objects by pressing “P key” after selecting the objects you want to separate. you can manually delete the edges that stayed at the cylinder.

For a two piece rocket and launcher

- **Method 1:** create a cone, rotate it, then extrude and scale to get the rocket shape.
- **Method 2:** create a cylinder, scale it along the Y-axis, then extrude one side, scale to about 1.3, then extrude two more faces, scaling the first about 0.3 and merging the vertices of the second at center.
- **Method 3:** Create a UV sphere with 4 rings, and model it - much in the same way as the penguin - into a rocket.



Place the sphere

just adding a UVsphere. The default 32 segments and 12 rings will be fine. This creates a smooth sphere.

You can think of the number of segments as being the wedges visible when the sphere is viewed as you added it. The number of rings then could be described as the depth of the sphere from that same view.

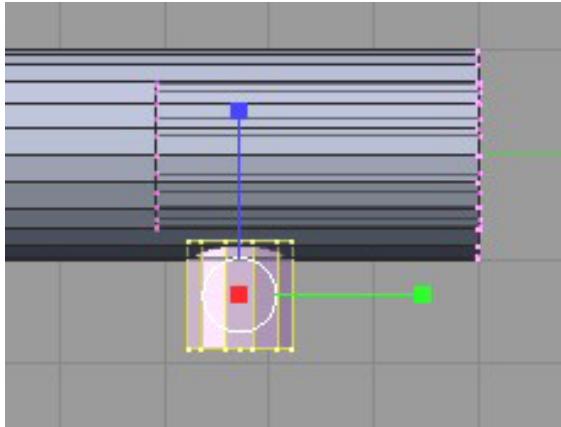
Resize and place the sphere at the new cylinder arm.

2.39.5 Create the mount

Having a launcher is nice, but we'll need to affix it to the jeep somehow. Let's add a mount to the tube.

Make sure you're in edit mode, not object mode!

Add a cylinder



Mount arm

Add a cylinder with 24 vertices with “Cap Fill Type” at “Nothing”, since we won't be seeing the ends.

Scale to about 0.55, then move it to the bottom of the tube.

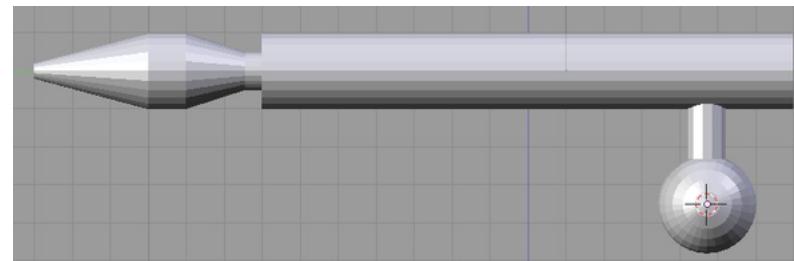
Add a UVsphere

The easiest way to have a wide range of motion for the launcher is to use a ball joint. We can simulate one by

Reposition the center point

This next step will be important for continuing the tutorial. Get the 3D cursor to as close to the center of the sphere as possible. While the sphere is still selected after creation, you can press SHIFT + S and snap cursor to selection, putting it in the exact center of the UV Sphere.

Switch to Object Mode. In the “Tool shelf” under “Tools” you'll find the button “Set origin”, click this and click on “Origin To 3D cursor” This should move the large pink dot where the cursor is located. This will give it a new center of gravity around the ball joint, making it easy to manipulate later.



2.39.6 Subsurf

Be in **edit** mode!

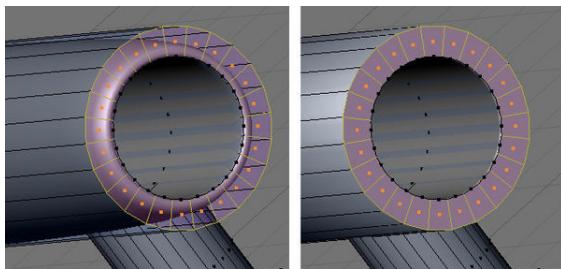
Apply subsurf, level 2

Select everything (A twice), and hit the **set smooth** button (at the bottom of the links and materials panel).

This makes the ends of the rocket launcher tube too rounded, as real ones are squared up.

Square up the tube edges on the right side

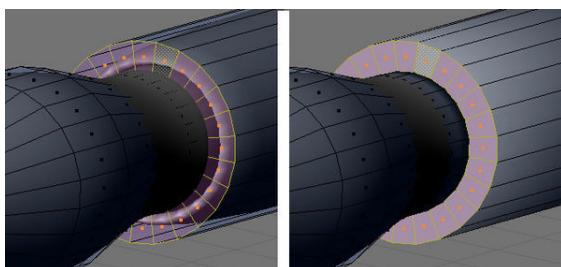
On the right side (without the rocket sticking out) be in **Face** select mode



RMB on one of the faces in the outer ring, then circle select (C), and select all the faces at the end of the launcher tube. You can also do a lasso selection by CTRL + RMB in face selection mode.

Crease the edges of the faces by hitting SHIFT + E and set it to 1.0.

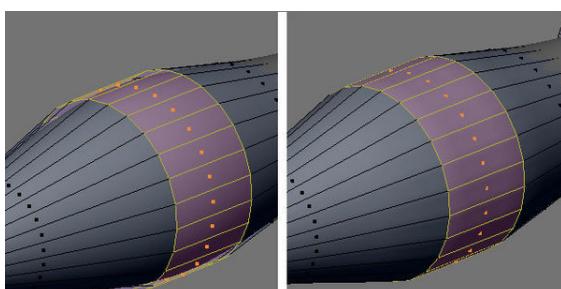
Squaring up the tube edges on the left side



Now do the left side exactly like the right side.

Noob Note Be sure to deselect everything on the other side first (A)!!

Squaring up the rocket



RMB on one of the faces in the middle surface, then circle select (C), and select the faces all the way around

Crease the edges of the faces by hitting SHIFT + E and set it to 1.0.

2.39.7 Final touches

Apply materials or additional items to the object and save for later use.

Do not forget to name the rocket for later use.

2.40 Simple Vehicle: Body

2.40.1 Techniques

You should already know how to:

- Make a mesh
- Navigate the viewport
- Extrusion
- Form faces
- Name objects

This section will recap and introduce:

- Deleting and creating edges
- Subdividing
- Merging vertices
- Loop subdivide
- Adding unconnected vertices in one object

2.40.2 Planning

The jeep is being designed to include the back, flatbed, door holes, dashboard, window, and hood. The window is extruded straight up (older jeeps' windshields aren't slanted), and I decided to add a lower back to later hold the bumper/lights if you wanted to add them.

2.40.3 Building the Jeep

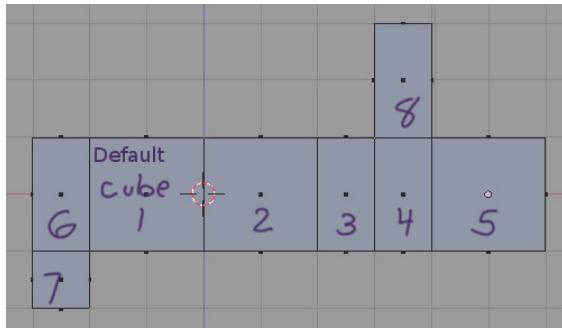
Extrude the Chassis

Start a new file.

Change to front view (NUM1), switch to **EDIT** mode, and deselect all (A).

Be in **Face** select mode

Starting from the default cube (#1), box select (B) on the right edge to select the right face, and extrude (E) 2 units (#2) to the right along the X axis (should default, otherwise X). Continue with three more extrusions to the right along the X axis of 1 unit (#3), 1 unit (#4), and 2 units (#5).



Main sections are made.

deselect all (A).

Now starting again from the default cube (#1), box select (B) on the left edge to select the left side face, and extrude 1 unit (#6) to the left in the X axis

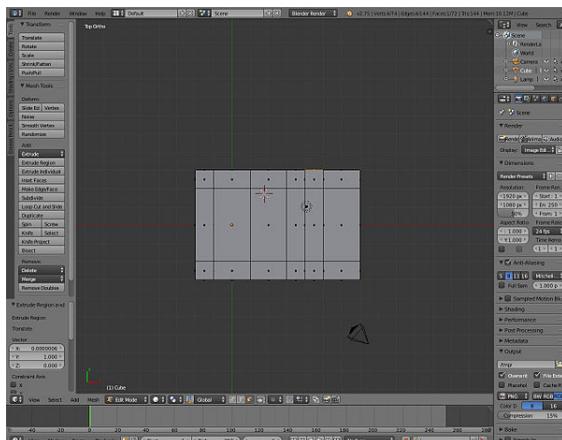
deselect all (A).

box select (B) on the bottom edge of block #6 to select the bottom face, and extrude 1 unit (#7) down in the Z axis.

deselect all (A).

Finally box select (B) on the top edge of #4 the second block from the right to select the top face, and extrude up 2 units (#8) in the Z direction.

Widen the chassis



Now to widen the jeep body.

Switch to top view (NUM7).

deselect all (A).

set “Limit selection to visible” off Increase the width of the existing boxes (#1) with one square (#2) by box selecting (B) everything (since you're in **face** select mode it will select all the faces of the starting chassis), now Scale (S) 2 times on the Y axis (Y)

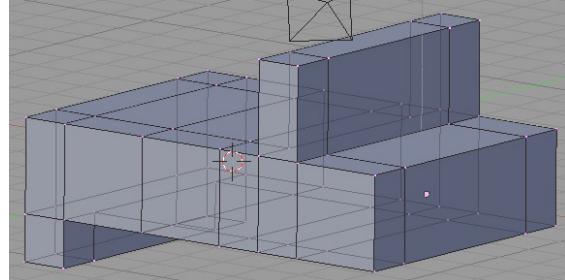
And select all faces on one side and extrude along the Y

Axis 1 unit. do the same on the other side

This should give you the figure that is shown to the right.

Noob note: holding CTRL as you do the above moves and extrusions will lock it to grid steps

Noob note: You might want to save your job here and rename it before carrying on as the options below are easier if you can revisit this stage.



Widening along the Y axis

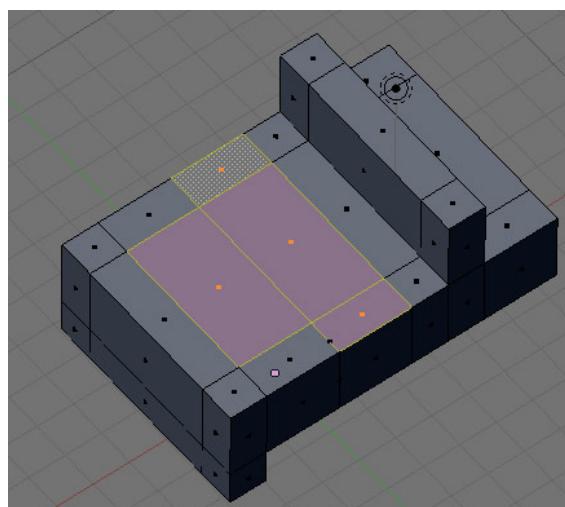
Flatbed and Doors

Our jeep design will have somewhat of a cheat - no actual door.

There are now a few different ways of doing this (thanks to reader submissions)! Try each one and study the results, as they will teach you about the issues you will encounter when you start making models of your own design;

Method 1 is the simplest of them, but teaches you the least in dealing with mesh trouble.

Method 1

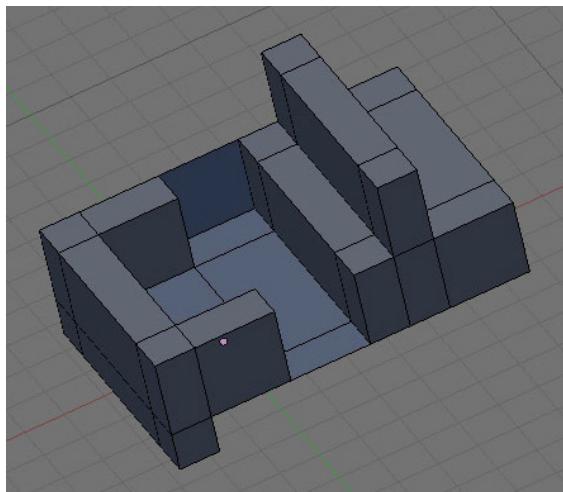
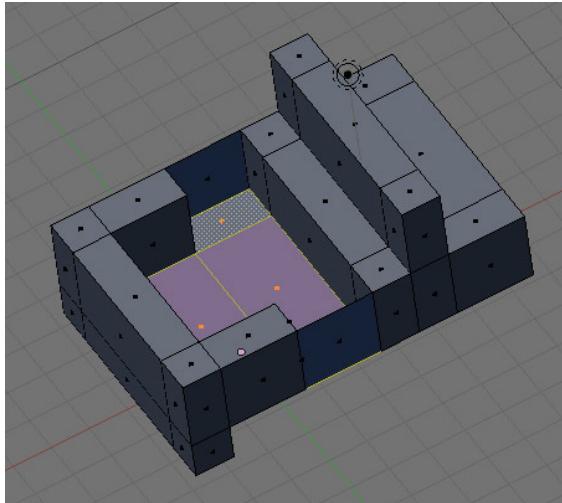


Make the jeep bed In NUM7 view

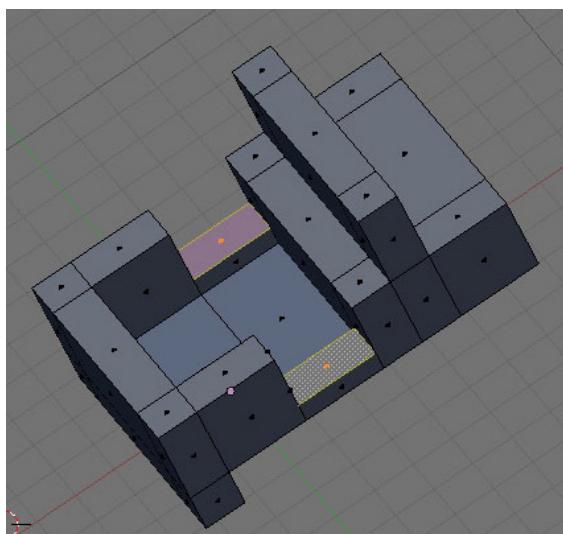
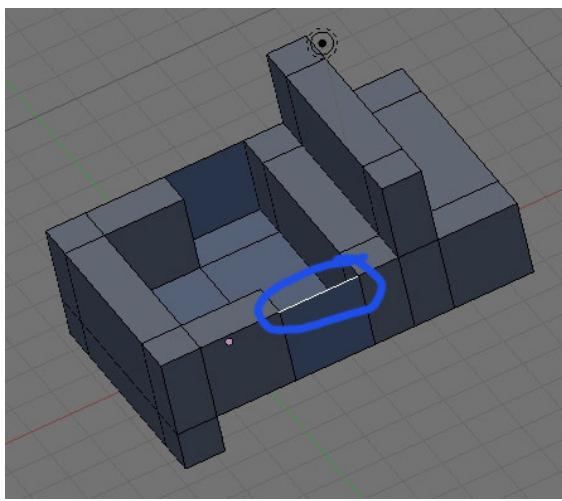
Be in **Face select** mode

Turn **occlude background geometry** (now called **Limit selection to visible**) on. It's the button with the spotty-box icon to the right of vertex, edge, and face select buttons.

Select (RMB , then SHIFT + RMB) the top faces where the flatbed and then doors should go. You should now have four faces selected, two large ones for the bed, and two small ones for the doors.



Extrude (E) the “region” –2, on the Z axis (Z).



Make the no-door holes Now in front view (NUM1)

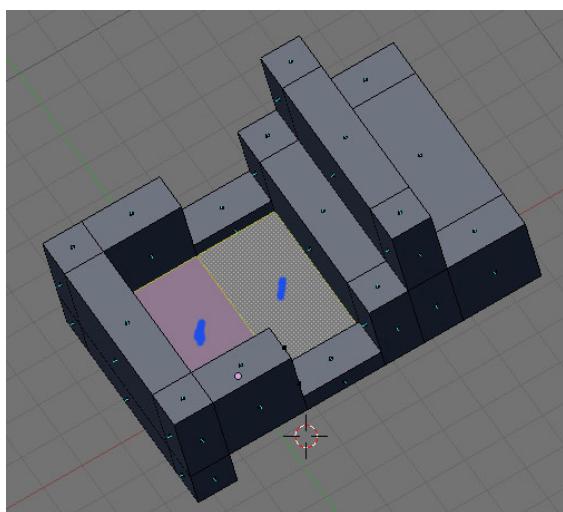
Be in **Edge select** mode

Select (RMB) the top edge of the door face, and delete (X) the “edge”. Next, do the same to the corresponding door edge on the other side.

Be in **Face select** mode

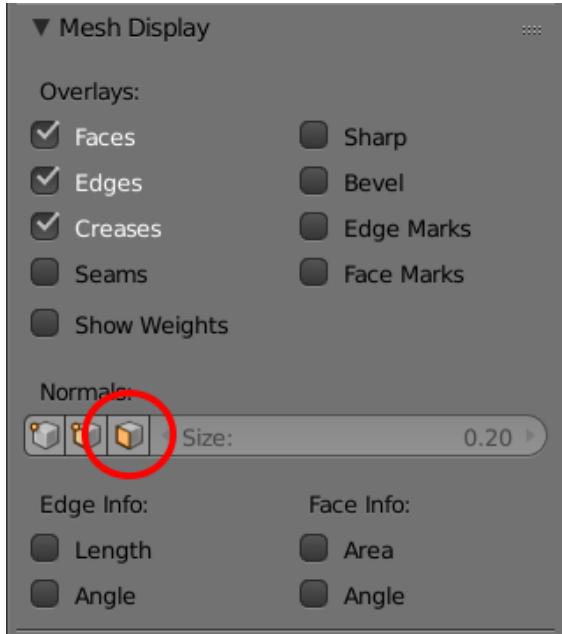
Select the faces in the bottom of the door wells, and extrude them up 0.5 along the Z axis.

Select the bed panels, and extrude them up .1 along the Z



axis (leaving them on top of each other will cause problems with rendering engines down the road)

Noob Note: Be very careful as you extrude the parts up, since they are on top of each other, it's easy to select the

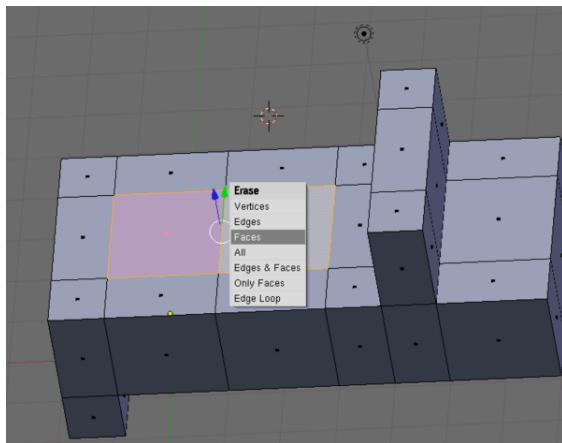


face on the outside, rather than the one on the inside. Once you've done the extrusion, check the normal lines by hitting the *draw normals* button in the mesh tools more panel (*if you can't see it, hover and scroll with the MMB*). If you don't get good normals, then hit undo, then reselect the faces, and try again until you get the correct face being extruded (50% chance of getting the wrong face pulled up). Or select the faces with a wrong normal and hit the *Flip Normals* button in the Mesh Tools tab.

Finally, select the whole body and remove doubles, (A) to select all then (W) to bring up the specials menu. On the special's menu, hit "remove doubles".

Note: removing doubles every couple of steps is a good habit to get into, and will save you time down the road.

Method 2



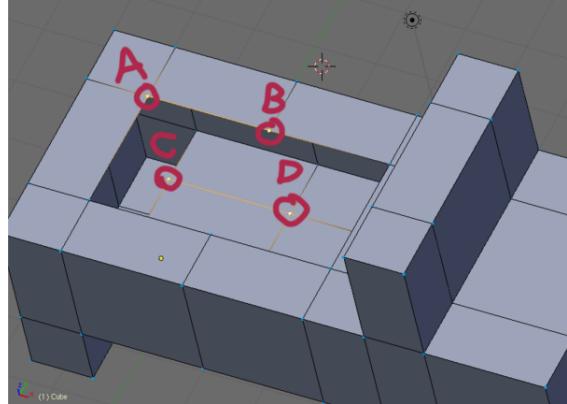
Deleting faces

Make the jeep bed be in Top view (NUM7), and **Face select Mode**.

Select the two faces in between the bumpers and wind-shield (RMB , then SHIFT + RMB).

Now delete both faces (X ->"faces").

Note: *you could also be in edge select mode, and delete (xkey) the edge between the two faces, for the same end result.*



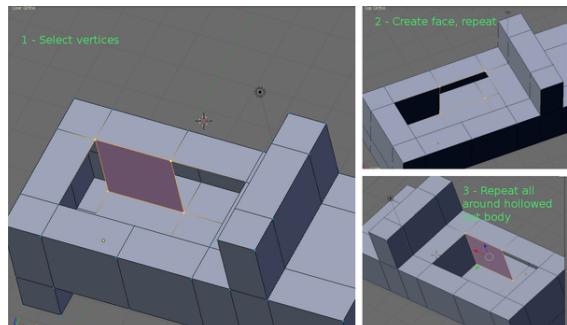
Selecting vertices

You will now add outer faces around the hole we just opened in the model, by selecting two vertices at the top of the model, and two from the bottom of the model.

Be in vertex mode

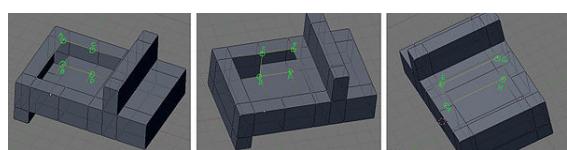
Deselect all (A twice)

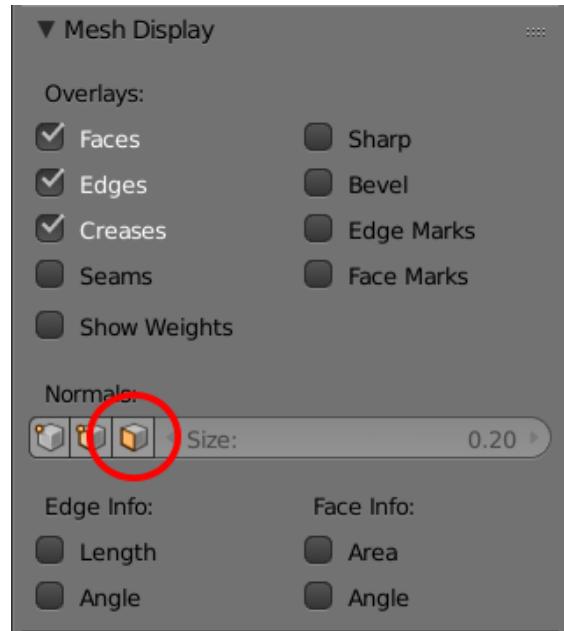
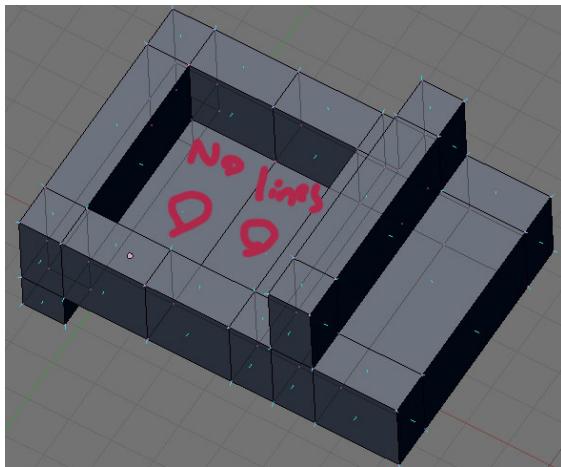
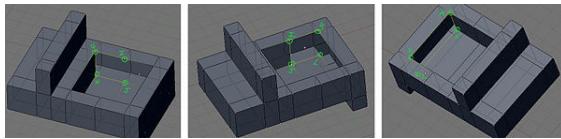
Starting at vertex A, circle select (C) vertex B, C, and D,



Creating faces

Next create a face (F); do this same procedure 5 more times at the vertx groups show in the pictures.





Click the button within the red circle to display face normals

Issues with exposed inside faces (Normals) Now that we have gotten the new outer faces in place we can't see into the model any more but we still have a big problem, the floor is actually the outside face of the bottom of the model, this is not good because you should never leave exposed inside faces (the other side of outer faces) on a model.

Exposed inside faces are invisible when looking through them when you apply textures and render it, since nearly all rendering engines completely ignore inside faces (A common problem with many sketchup models, ah the irony!).

You can tell a face is an inside face, by the fact that if you turn on show normals, and show vnormals, on the mesh tools more panel, the outside faces and vertices will have blue lines shooting out of them, while inside faces will not.

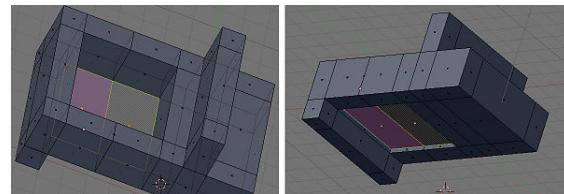
Understanding this now will save you lots of wasted hours in the future trying to fix your normals.

Press **NKEY** and go to “mesh display” and click on the cube with the face highlighted. Look closely at the exposed interior faces in the floor of the truck bed, they don't have blue lines popping out in the inside, only on the outsides.

Now select textured from the box where you pick wireframe, or shaded, and the exposed inside faces will become invisible when viewed from the inside, but look great when looked at from the outside.

In newer versions it is no longer invisible, and it seems to work but it still has no normals, and it is still important to know.

Add the floor of the bed Lets make a floor for the bed now.



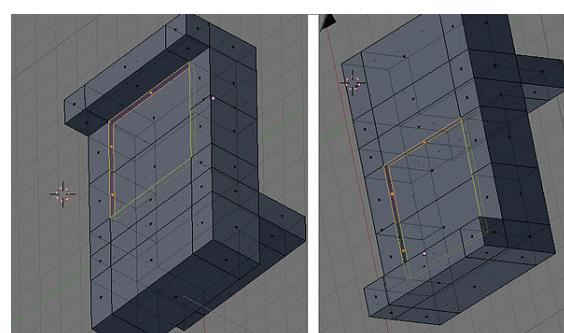
Be in **Face select** mode

Select (RMB) both faces of the truck bed,

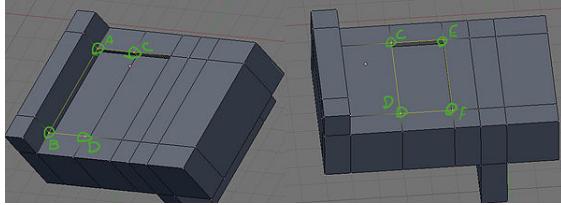
Extrude (E) pick “region” from the popup, and move up along the Z axis (Z), by about 0.2.

With the two faces selected, hit mesh -> normals -> flip to make their normals correct for their new position

Now rotate the model till you're looking at it from the bottom, as you can see there is now a hole that needs some extra inside faces removed, and new faces added to complete the bottom.

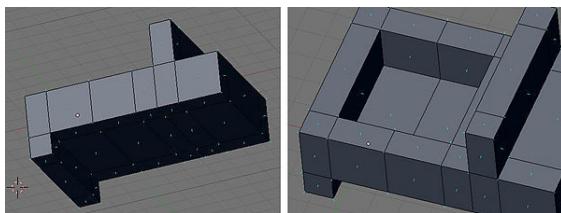


Select each extra face on the sides, and delete (X) the "faces" as shown in the photos.



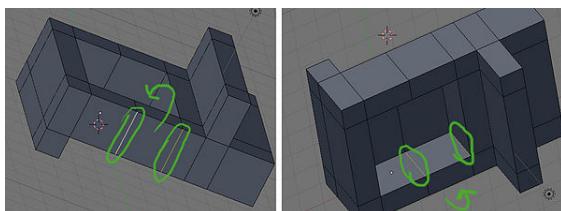
Change to Vertex select mode

Select (RMB) corners a, b, c, d, and create a face F , then do the same for c, d, e, and f.

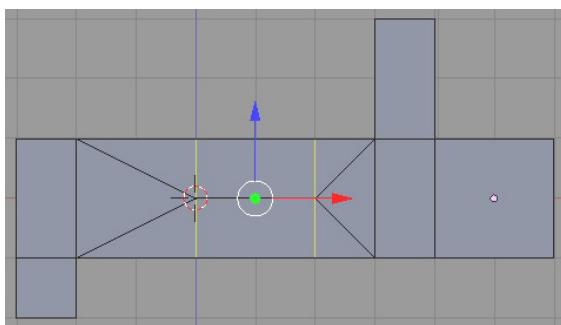


Recheck the normals, and make sure you only have outer faces on the outside of your model.

Make the door holes Method 2.2 is easier!

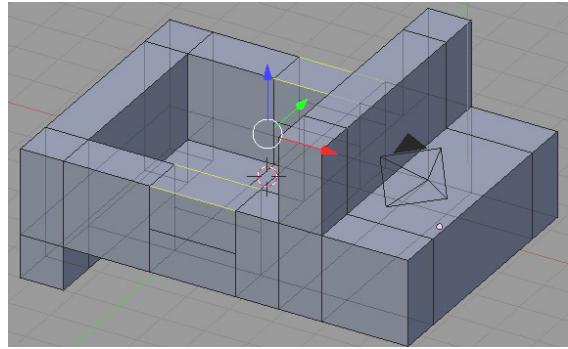


Method 2.1 Subdivision To cut out one door, select the four vertical edges where the door will go.



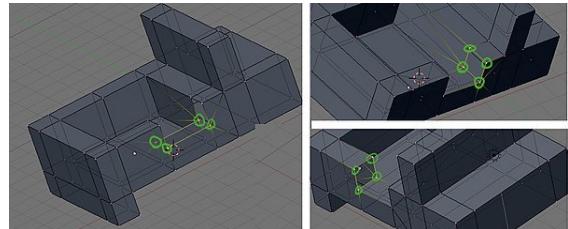
Split the area where the door will go.

Use the subdivide command (W) to cut the edges in half. You'll notice that the subdividing will also affect the adjoining faces.



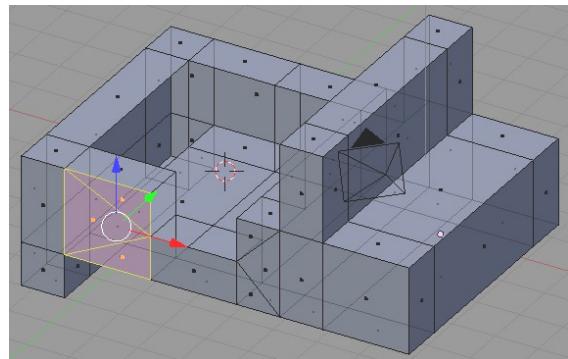
Delete edges for door opening.

Select the two edges at the top of the door panel and delete them, removing the top half of it.



Change to Vertex select mode

Now build up faces by selecting the groups of vertex's as shown in the photo, and creating a face (F). Do the same thing for the other side, removing/adding edges and creating new faces.



Merging faces.

If you want to clean up the look of the sub divided faces in the model

Change to **Face select** mode, and select the faces that are going to be combined.

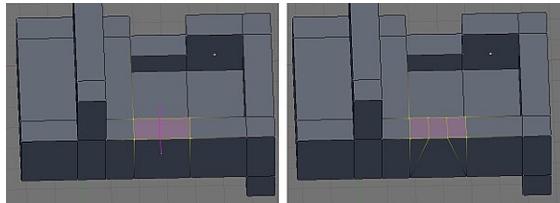
Press F and a little window will appear titled 'Make Faces'. Click on 'Make FGon' to merge the faces. (As you may have guessed, if an FGon face is created and you want to later undo it, select the 'clear FGon' option in the Make Faces window.)

Note: to merge an FGon to a flat face, select it and hit CTRL + J

Pro Note: You don't even have to make an FGon. Just select the door-to-be, Tris and Quads both, and hit **CTRL + J**.

Noob Note: Q. What is an FGon??

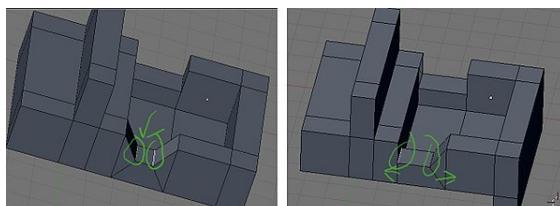
A. It is a fake polygon, a way to hide triangles and quads on flat faces.



Method 2.2 Multicut

Be in frontview (NUM1). Select the top edges where you want the doors

Multicut (KKEY), enter two in the popup, draw a line along the axis of the cut, as shown in the photo, and hit **ENTER**.



Then move the newly created edges down and towards the sides.

method 2.3 (Noobie)

Select top face of where door is going to be

E to extrude, pick "Region" from the popup menu Z to lock in Z direction hold **CTRL** to snap to whole BUs and move down one BU

Select 2 top edges that look like they have no widths

X to erase, pick "Edges" from the popup menu

Two faces will be missing. For each individual missing face, Select top and bottom edges

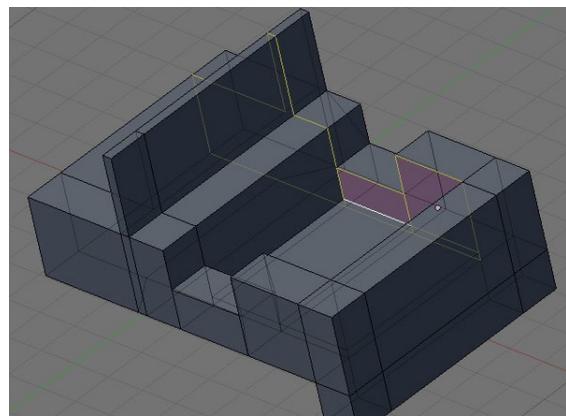
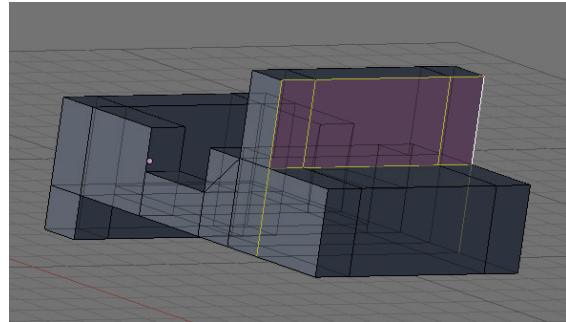
F to fill in the missing face.

Resizing the bed and windshield

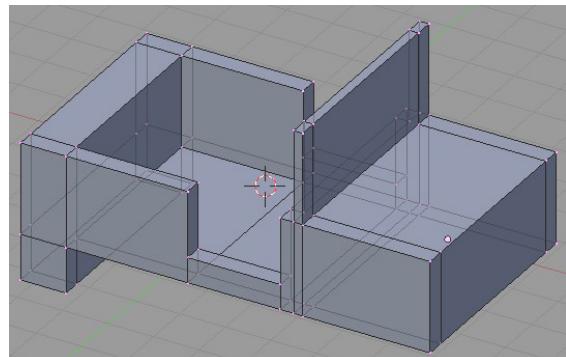
Go ahead and resize the widths of the bed and windshield.

As a precaution, Remove doubles before starting!

To narrow the width of the window, change to **Edge mode** and select the edges shown in the picture, then move (G) along the X axis.



To narrow the width of the side, change to **Edge mode** and select the edges shown in the picture, then move (G) along the Y axis.



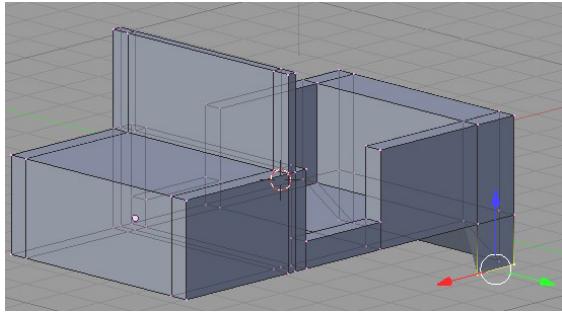
Adjusting the widths.

Always move as many vertices or edges at the same time to not only work faster but to make sure they are moved equally. The use of circle select, loop select, and SHIFT while moving vertices is very helpful in fine movements.

If you want an object to come to a point such as a wedge from a cube, merge vertices.

In this example, the lower back area will be modified. Select two vertices to join together and press **ALT + M**. Select the option for your merging. 'At First' or 'At Last' will probably be the option that will work here.

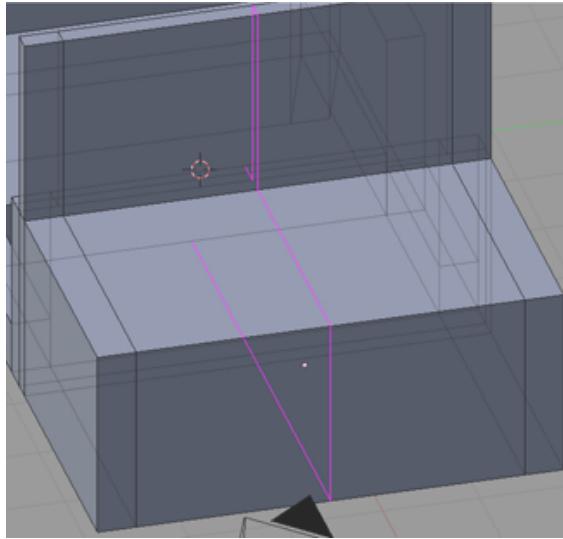
Play around to see what each merge option does. After the merging, Blender will tell you how many vertices were



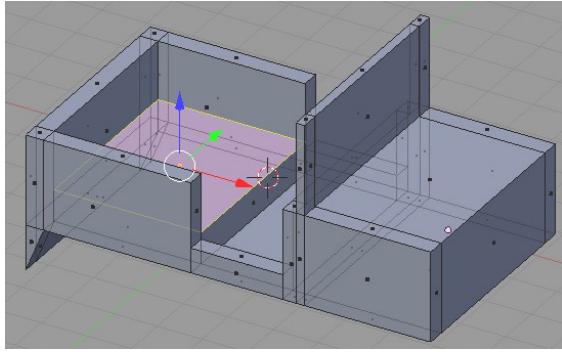
Bringing the back to a point.

removed.

Be certain to “remove doubles” (W), as merging creates lots of them. (You can also find it on the menu Mesh ==> Vertices ==> Remove doubles)

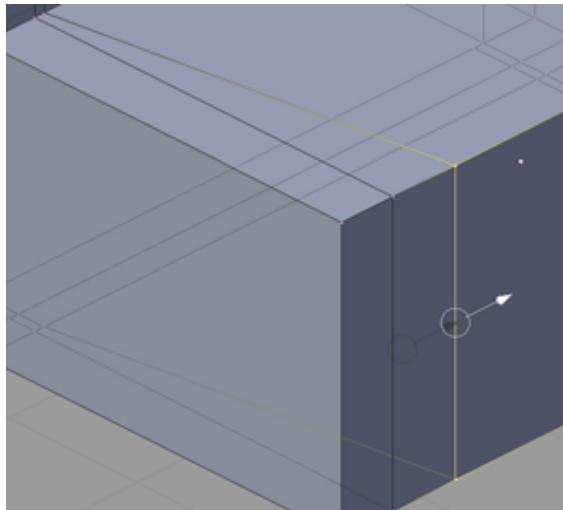


Loop Subdivide select.



Raise the flatbed.

Extrude the bed surface upward. This is only useful in hiding the tires that we'll add later in the tutorial. Alternatively, you could make two boxes to hide them.



Aligning the hood.

2.40.4 A Touch of Detail

Adding an Engine Hood

Method 1 Let's add some detail to the model - how about the hood? First thing to do is add some edges to the front of the jeep. Press CTRL + R to enter Loop Subdivide. A pink loop will appear around the mesh. Put the cursor over the area to get the example picture to appear.

When the loop is in the right place, LMB click. The place to put the actual cut can now be selected. Do this twice - once for each side. Move and align the resulting edges to form an angle to the front and bring the window vertices in.

Extrude the hood surface up a small amount. We don't want it too high, just high enough to catch the light.

Zoom in and select the top-front hood edge created from the extrusion. Drag it out along the X axis. Select the now diagonal face of the hood extrusion. Extrude from it. The result will come out of the surface at a diagonal

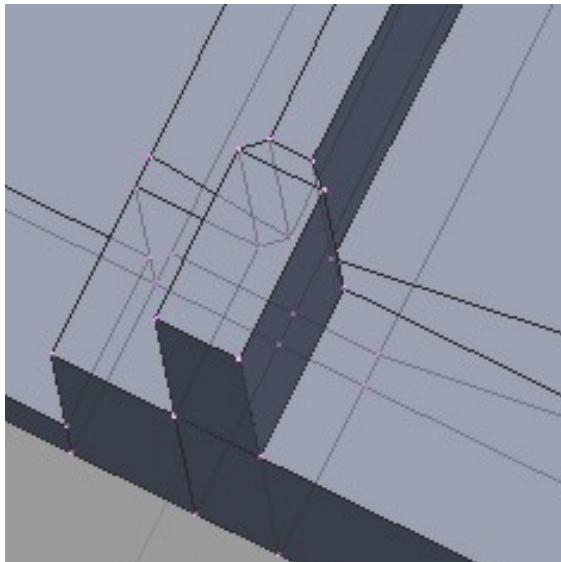
angle. Take the resulting vertices and move them close to the front of the jeep.

Using the additional lines from the loop subdivide you can improve the shape of the window

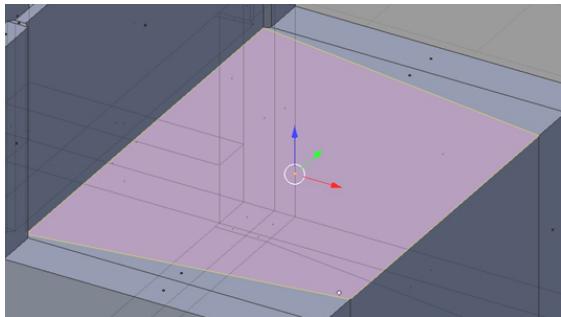
Method 2 Another way to raise the Jeep hood. It doesn't require loop cuts, so is a bit simpler.

Step 1. Make Inner Square.

Switch to Top-view (NUM7), or slightly rotated off for easier viewing. Face-select the top square of the hood. Do E -xtrude, then hit <esc>. NOTE: this WILL make a new surface, hitting ESC doesn't cancel the extrude, just makes its location to be exactly on the old surface. Do S -cale and type 0.9 and hit <return>. Now you will see the new surface as a smaller square (or really rectangle) on top of the jeep hood square.

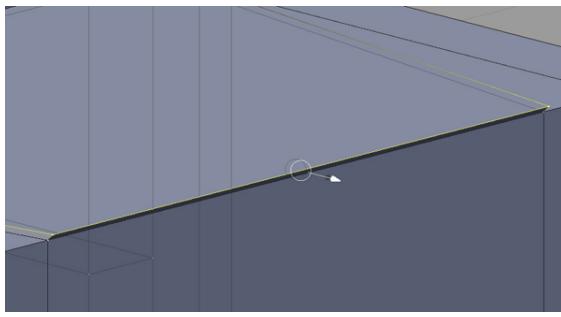


Formatting the window.



Hood extrusion.

Noob Note: What I had to do is S -cale on Z and type 0.9, otherwise the New Surface For the hood would end up shooting through the Windshield.



Bring out the top front edge.

Step 2. Shape Hood.

Still in Top-view NUM 7, deselect all.

Box-select the right two vertices of the new square, towards the front of the jeep.

Do S -cale on Y, and type 0.8 and hit <return>. Leave

the two vertices still selected.

Do G -rab on X, and type 0.2 and hit <return>.

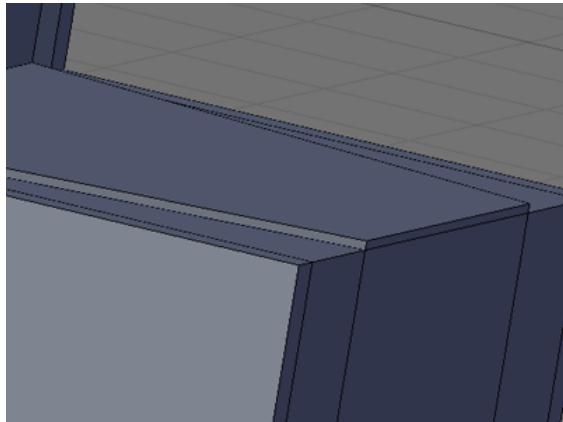
Face-select the resulting quadrilateral.

Switch to Side-view NUM 3.

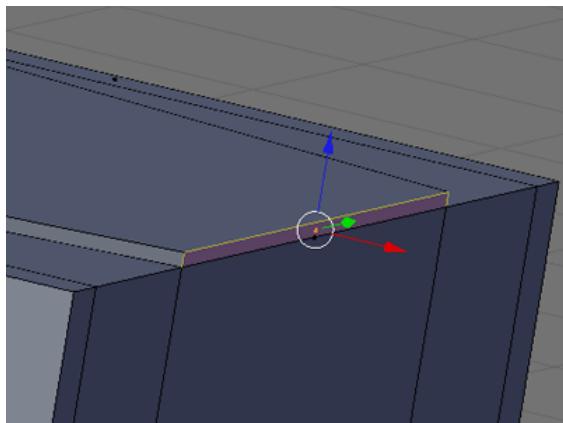
Do E -xtrude on Z and type 0.2 and hit <return>. I used 0.2 to exaggerate the screenshot a bit, you probably want 0.1 instead.

Now you should have a raised hood on the front of the jeep.

method 3 Another way to extruding the Hood of the jeep including a lip that comes over the front.

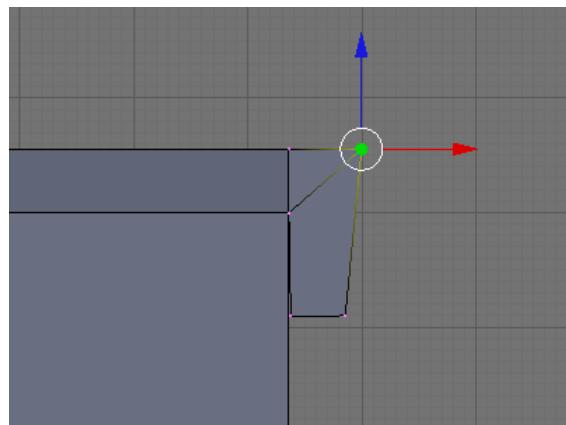
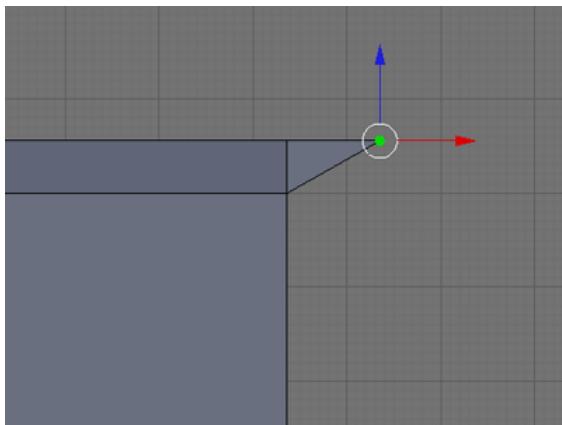
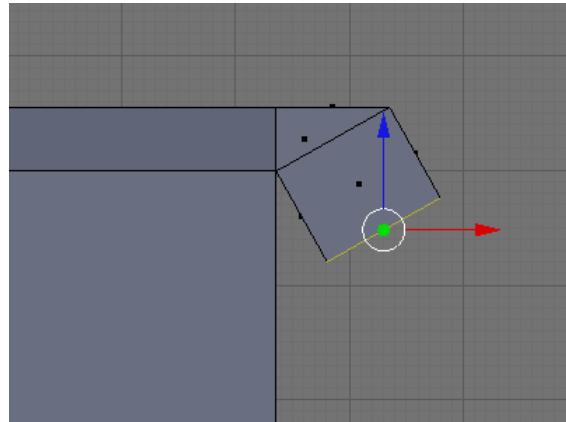
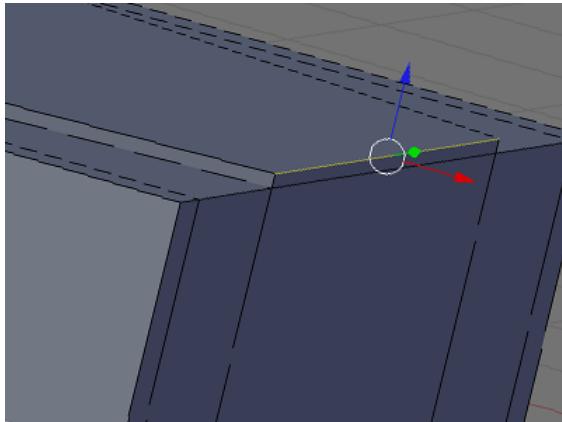


Step 1: In face select mode (CTRL + TAB ->faces), select the top of the hood and press E to extrude it. Only extrude it a small amount.

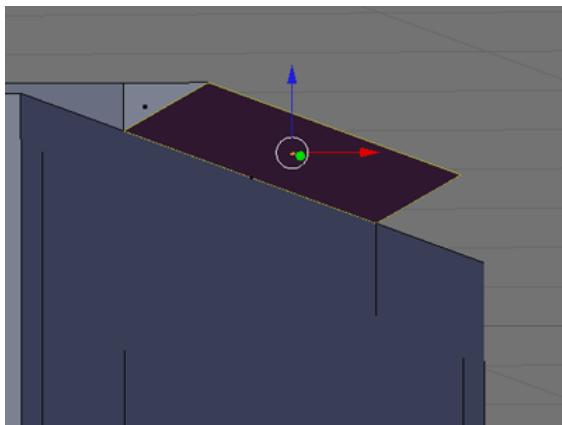


Step 2: Staying in Face select mode, here is the tricky part. Select the small face on the front of the hood you just extruded. Then, hit E then ESC . This creates a duplicated face on top of the one you selected. Do not click or move the mouse between these two keystrokes.

Step 3: Now deselect the selected face by hitting **AKEY**. Then enter Edge select mode (CTRL + TAB ->edges) and select the top edge of the face you just deselected.



Step 4: Hit NUM1 to go to the side view. Now using the red X arrow pull the edge out a little further than you pulled up the hood itself then hit LMB .

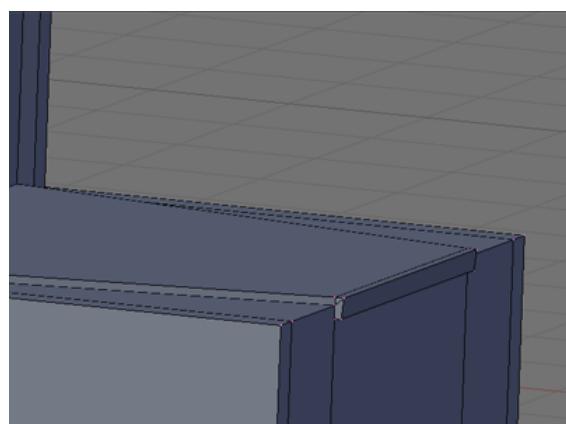


Step 5: Now rotate just enough so you can see under the wedge you just made. Go back into Face select mode (CTRL + TAB ->faces) and select the face on the underside of it.

Step 6: Go back into side view with NUM1 . Now hit E to extrude the face a little with the mouse. When it's a good size hit LMB .

Step 7: Now make sure the “Select only visible” button is

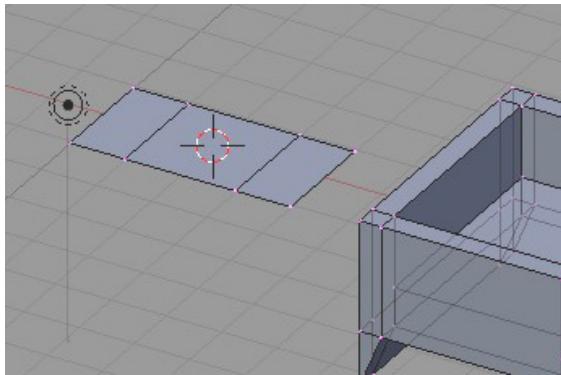
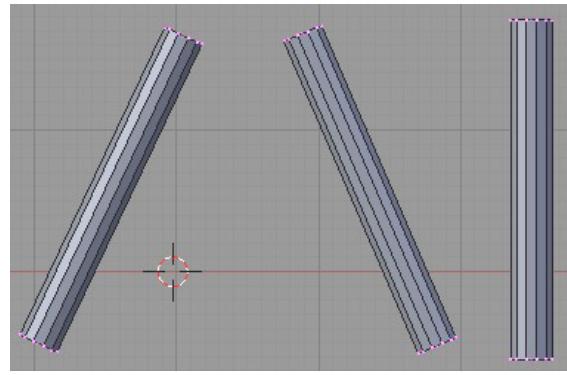
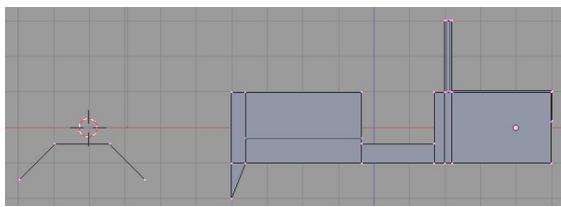
turned off (“Occlude background geometry” in later versions) and go into Vertex select mode (CTRL + TAB ->vertices). Play around with the vertices pulling them a little closer to the front of the jeep. It's best to select using the box select (B) or the Lasso (CTRL + LMB).



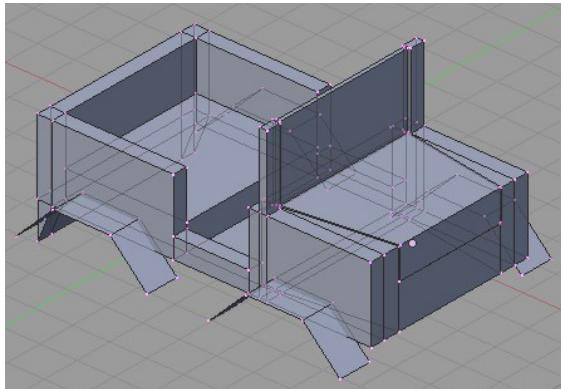
The end result should be an extruded hood with a lip.

Add a Fender

In the topdown view (NUM7), add a plane. Extrude an edge twice to result in three connected planes.

*Starting the fender.**Setting up the tripod in NUM1 view.**Bend the fender.*

Pull the sides down to form a trapezoid shape and reduce the width.

*Model with fenders.*

Once it is in the desired shape, select the three faces and duplicate it. Press SHIFT + D and all selected vertices, edges, and/or faces will be duplicated. The copy will automatically be grabbed for moving.

Move the duplicate to the jeep body and repeat the duplication two more times for a total of four fenders.

Noob note: A good idea is to first position one fender, then copy it and restrict movement to x or y-axis. Then copy both fenders and move the two new copies along x or y-axis. Much simpler than trying to position four fenders individually.

Add a rocket launcher mount

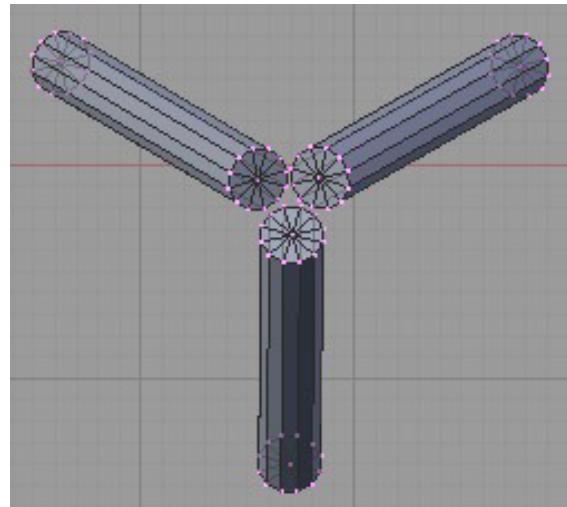
We'll move on to making a tripod support for the rocket launcher.

Add a cylinder mesh with 12 vertices then scale and size it so that it looks like a tube. Once you have it to a size you like,

Duplicate it twice for a total of 3 cylinders.

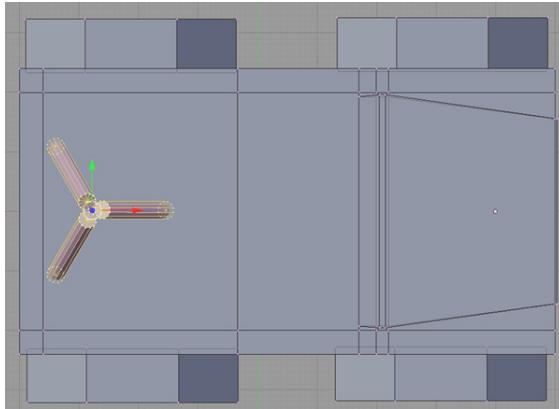
Rotate two of the cylinders in the NUM1 view by LMB clicking on the white circle that appears when the cylinder is selected in rotate mode.

The picture on the left is an example of the end result.

*Setting up the tripod in NUM7 view.*

Change to overhead view (NUM7) and put together the three cylinders so the tops come close together. Now all three can be selected and moved or rotated accordingly.

Move the tripod onto the jeep flat bed. The final steps are to select your materials and rename the object (described in the wheel section). This will complete our simple jeep model.



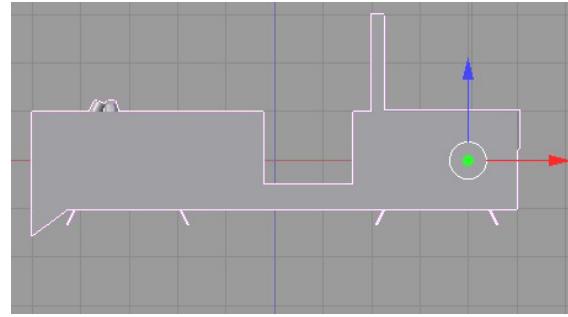
Putting in the tripod.

2.40.5 Subsurf

Since we expect a jeep body to have squared up edges, we won't subsurf this part of the model.

2.40.6 Optional Activities

Feel free to add anything you see fit such as bumpers, guard rails, doors, steering wheel, lights, etc. You can either have them on the same object or separate objects (useful if you want to move them around).



Body object

In Object Mode, go up to **File > Append** (**Append or Link** in later versions), about 3/4 of the way down the menu. The Find file window will appear. Go to the location where the jeep seat was saved. When the .blend file is clicked, you'll go into it as if it is a directory.

Here we have the categories of Camera, Lamp, Material, Mesh, Object, Scene, Text, and World. We are interested in the seat object, so click on Object. Now there are three items: Camera, Lamp, and Seat. That is, it will say Seat if you named your object Seat. This is why it is useful to rename your objects, materials, etc. If you forgot to rename the object, it will be called Cube (default for our starting mesh).

Noob note: You must be sure you're appending, and not linking! If you try to duplicate it, and you get an error, then it's probably linked, there will also be "li" to the right of the ME: object name button. Ways to be certain you're appending are: when selecting the file, look at the bottom for the append/link options and make sure "Append" is selected; In the file menu, don't choose "Append or Link (Image Browser)", you need to use the option above it.

2.41 Simple Vehicle: Some Assembly Required

2.41.1 Techniques

You should know how to:

- Do everything discussed in previous tutorials

This section will recap and introduce:

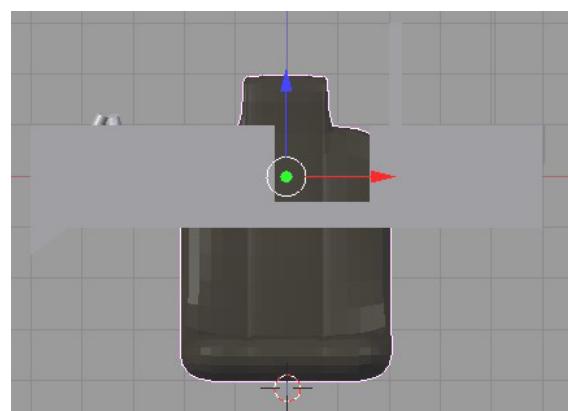
- Append a file
- Duplicate an object

2.41.2 Overview

The objects for the simple vehicle have been made if you have followed all the previous Simple Vehicle tutorials. Putting it all together will come very easy now.

2.41.3 Appending the File

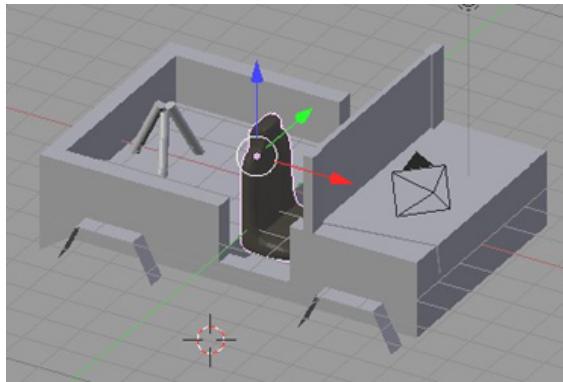
If you have the jeep body file open, keep it open. Otherwise, open the file for the jeep body.



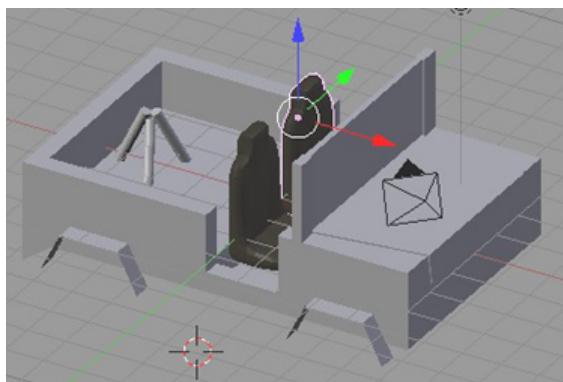
Appended seat object

What should happen after selecting Seat and the button 'Load Library' is the seat will pop into our file where the 3D cursor was.

It will definitely need to be scaled, rotated, and/or moved to the right position. One way to do this is to rotate the seat about the Z axis -90 degrees by pressing R Z 9 0

*One seat*

NUM- .

*Duplicated seat*

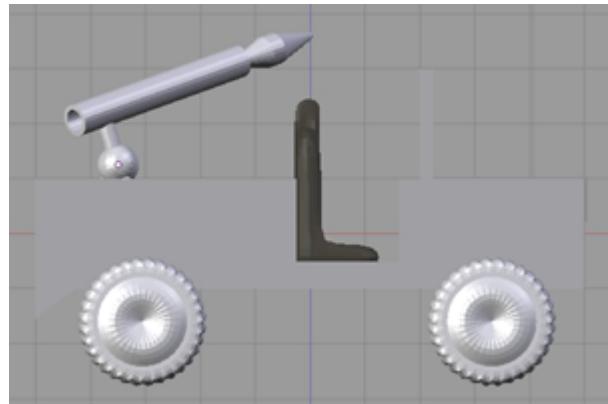
After placing the seat in the jeep body, let's make another so we have a seat for the driver and passenger. Still in Object Mode with the seat selected, duplicate it (SHIFT + D) and slide it over. After duplicating it, it will automatically go into grab mode. If you RMB or ESC , it will still be duplicated - just sitting on top of the original.

Noob note: Duplicate appended objects in object mode, not edit mode, as they are separate objects and do not share the same edit mode space (each appended object has its own edit world, unless you join them together)

2.41.4 Rinse and Repeat

Append the file again to place the wheel object and rocket launcher in the file. Scale, rotate, move, and duplicate each object accordingly. Depending on the position of your camera, you may or may not have to make all four tires. Remember that the only important parts to draw are those that will be seen!

Set the ball over the tripod. The fun part is rotating the rocket launcher since the center of it has been moved to the ball joint.



2.41.5 Parenting

Be sure to parent each item to the jeep chassis, by selecting the item, lets say the tire, then also selecting the chassis (RMB then SHIFT + RMB) and then CTRL + P , and select "make parent" from the popup menu.

2.41.6 Final things

*Rendered jeep*

The last thing is to apply materials to your objects!

You can apply material in either object, or edit mode. As you might have noticed when you select something in object mode the entire object is selected, in edit mode on the other hand individual faces can be selected, and painted.

Make the windshield look like glass

You have to create a new material and assign it to the appropriate faces.

In edit mode, select the faces you want (make sure you select both sides of the windshield including: front face, back face, and top face).

On the Editing section F9, under Links and Materials, click the 'New' button under Materials then click the 'As-

sign' button. This assigns the new material to the selected faces.

You can then go into the Shading section F5 and adjust the material to be glass, by setting the alpha to .20, and hitting the ray transparency button. Three more ways of making a material that looks more like glass are shown in Material Glass.

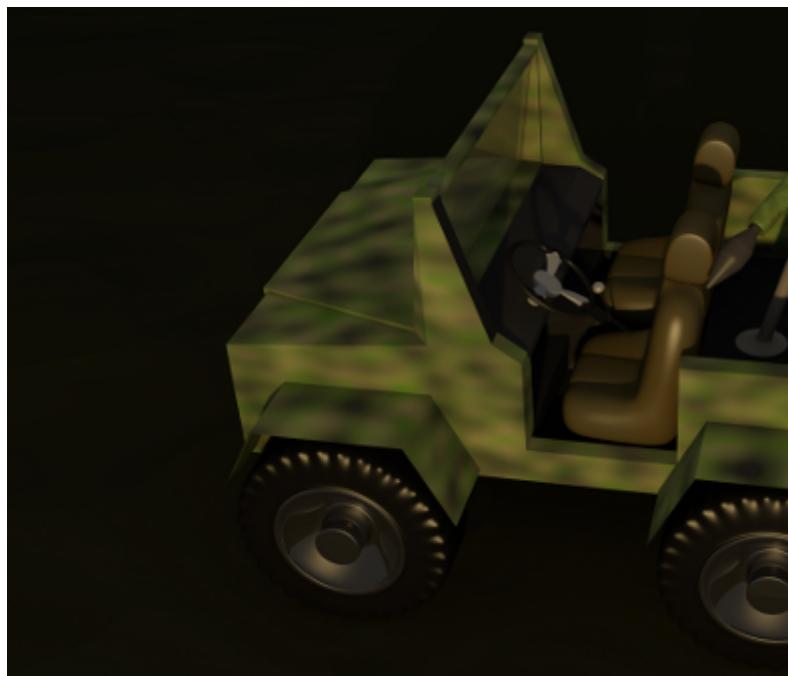
Make sure you are editing the correct material (if you have just the one texture on the jeep, the default name for the new material should be something like "2 Mat 2").

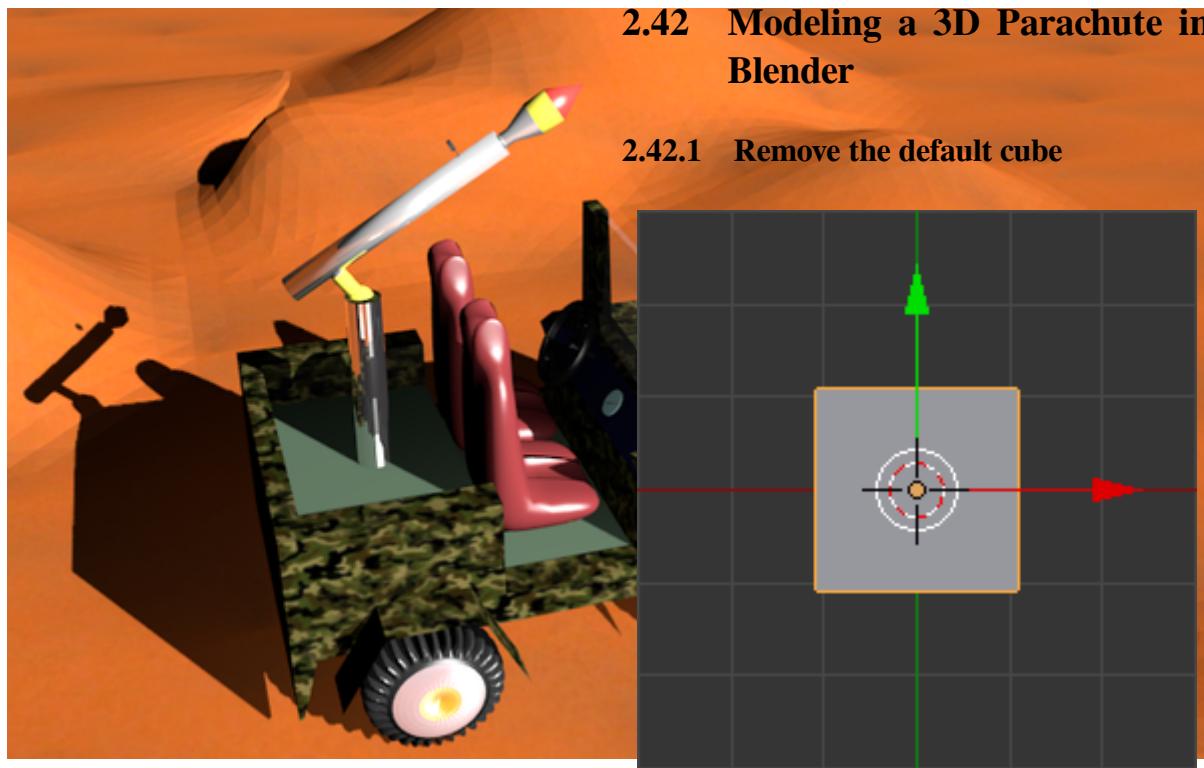
Noob note: Glass is only transparent when you render it. in order to render it you will need to position a few lamps around the jeep, and move the camera around so that it can see the jeep (view->camera).

2.41.7 Extra

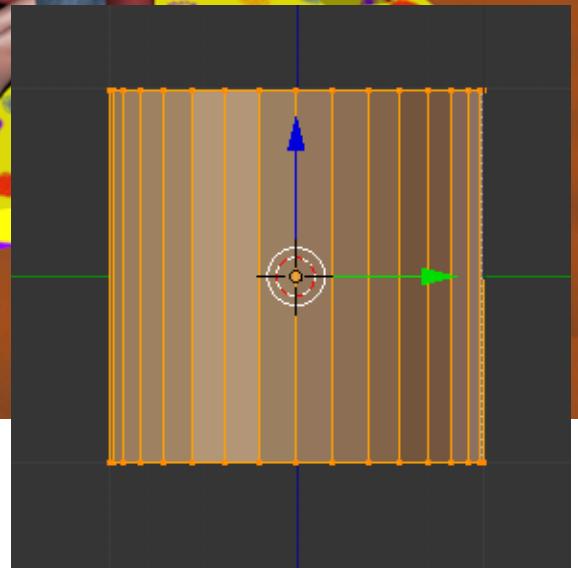
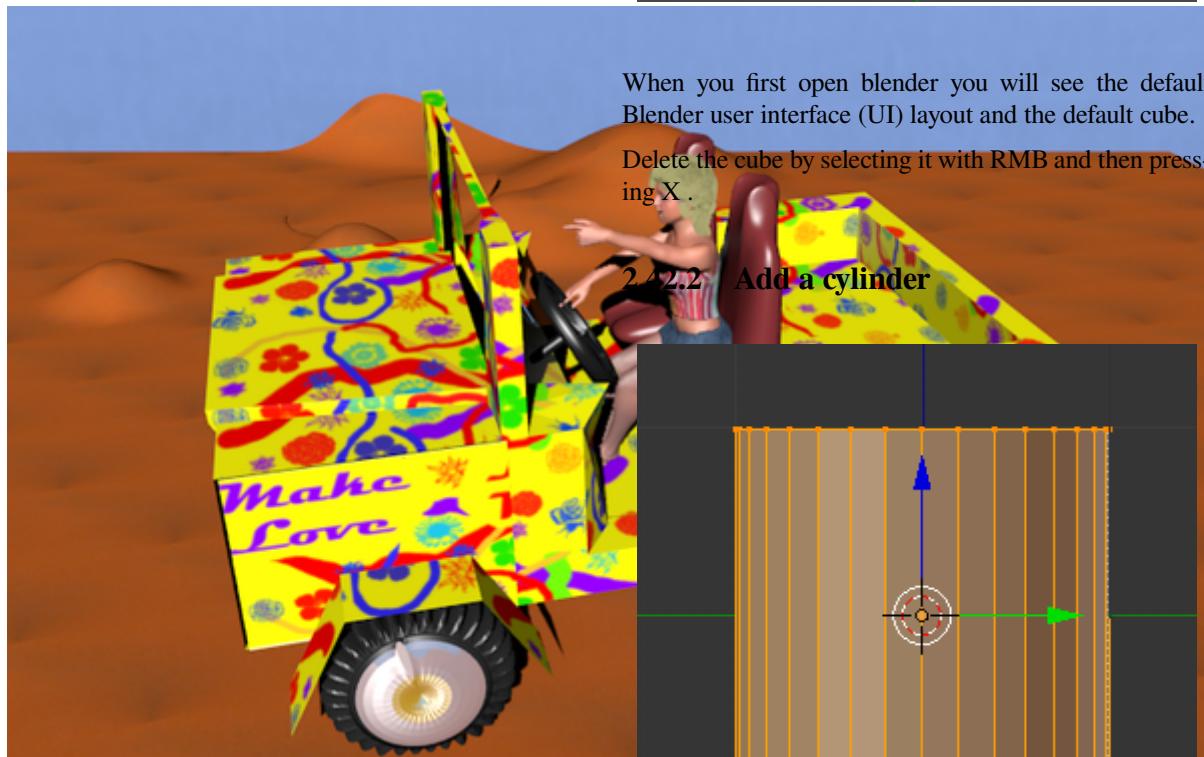
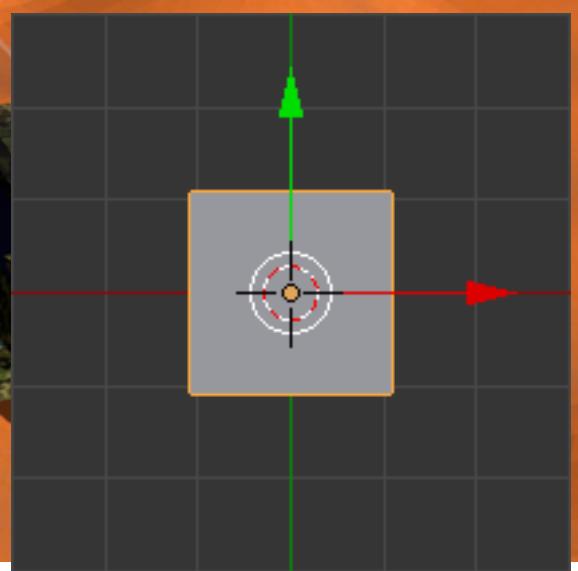


Other possible ideas





2.42.1 Remove the default cube



Jeep | Render | Version 5

2.41.8 Additional Tutorials

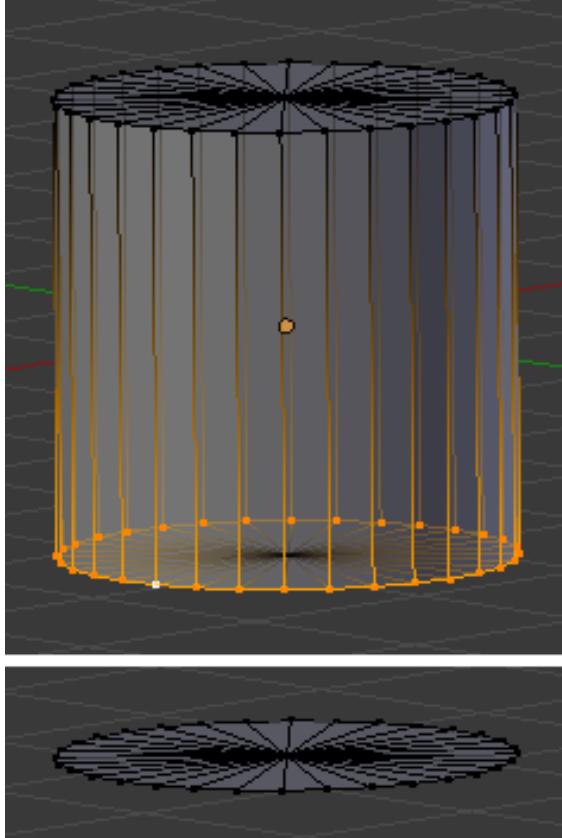
Append and Link Video Tutorial: <http://www.youtube.com/watch?v=69ZBIDrOlIY>

Switch to **Side** view (NUM3).

Add a Cylinder with SHIFT + A → Add→ Mesh→ Cylinder. At the bottom of the Tool Shelf change “Cap Fill Type” to “Triangle Fan”.

Press TAB to switch to Edit mode. The cylinder should now look like the picture at right.

2.42.3 Remove the bottom row of vertices



Next you will want to delete the bottom row of vertices. To accomplish this go into side view by pressing NUM3 .

Ensure that you are in **Vertex select** mode (that the left-



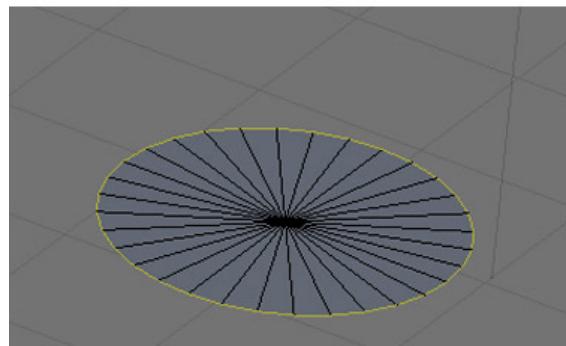
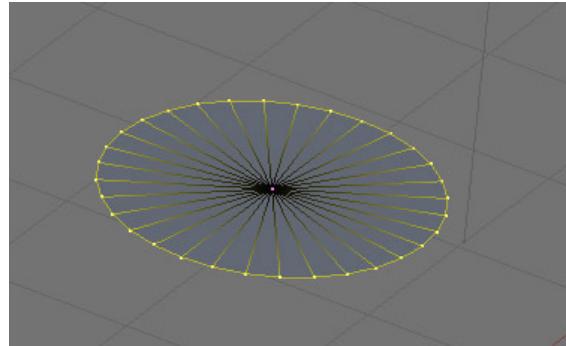
most of the group of 3 icons is selected; click on it if not).

Press A twice to deselect everything.

Press B , and box select the bottom row of vertices, and press X to delete and then confirm.

Note: you must have limit selection to visible off, the button to the right of the group of three icons for selection. If you wish to do the selection with it ON, you will have to change your perspective so as to be able to select all lower vertices.

Note: Instead of all of the above, you can also just add a circle with fill type 'triangle fan'



2.42.4 Extrude and scale to shape

Be in **Vertex-select mode** (as above).

Select all the vertices around the outside of the circle. The centre vertex must not be selected. You can do this conveniently in a number of ways; why not practise them all:

- Starting with no vertices selected, use A to select all vertices, then SHIFT + RMB on the centre vertex to deselect it. Or
- Starting with no vertices selected, ALT + SHIFT + RMB (*loop select*) on one of the outside edges or vertices to select the entire loop of them. Or
- RMB on the centre vertex to select only it, then use CTRL + I to invert the selection.

Now switch to edge-select mode by CTRL + TAB and selecting Edges.

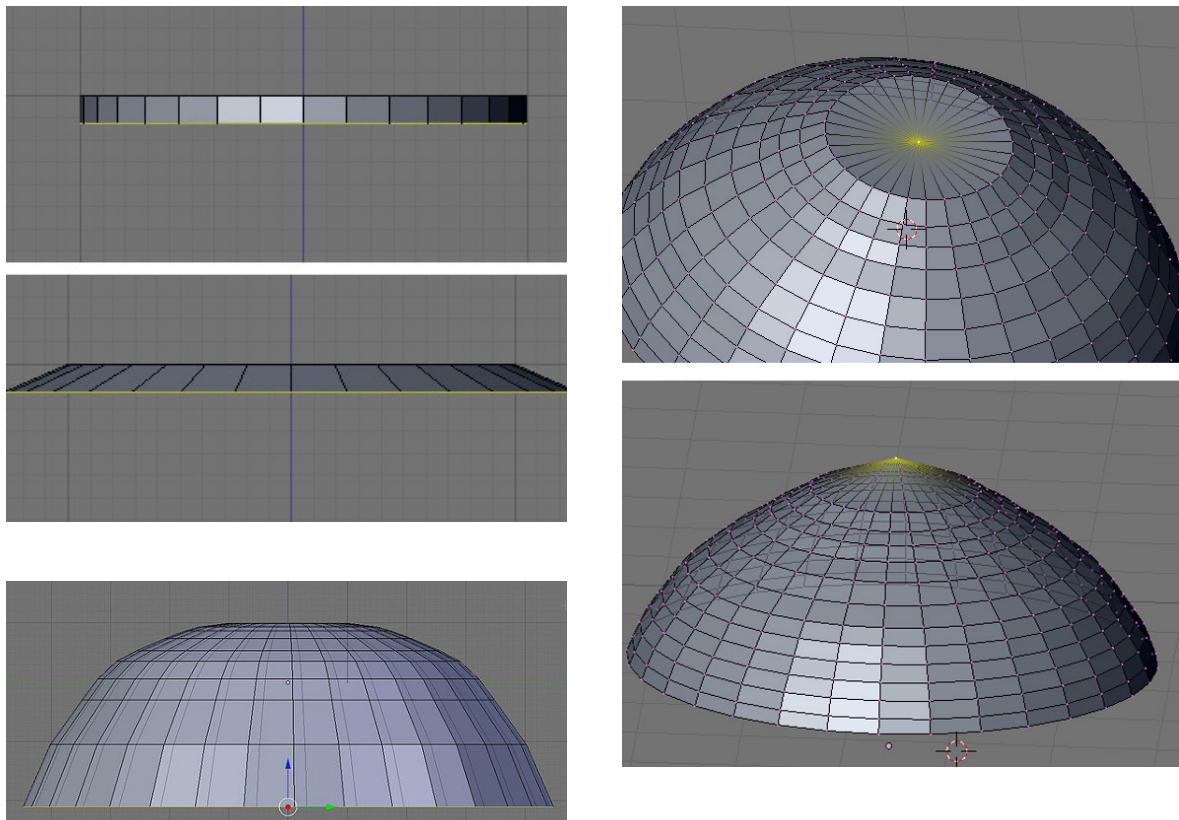
Press E to extrude ("region") the selection and Z to constrain to the Z axis.

Drag the edges a small ways down then click LMB to release.

As this makes a right-angle at the sides and parachutes don't have straight edges, we need to scale the selection outward.

Press S and move the mouse away from the model, you will see that the edges get smaller and bigger.

Scale them out a small ways then left click to release. You will want to practice a little with the scale amounts till you can make a realistic parachute shape.



Continue extruding and scaling till you have a shape like the one shown on the right.

When you're done making a nice shape, be sure to select all (A twice), then W and pick "Remove Doubles".

2.42.5 Make the top more rounded

Be in **Vertex mode**.

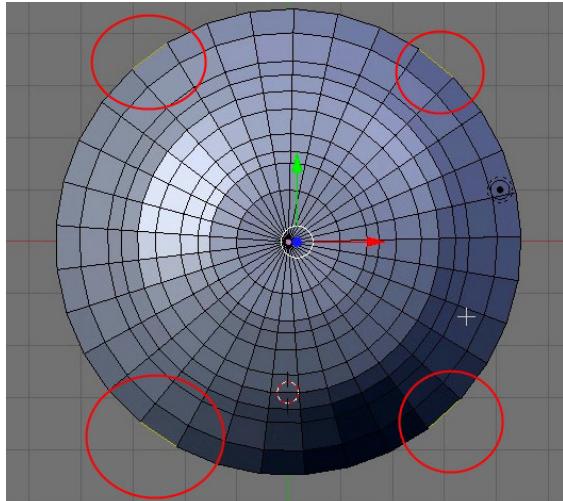
Select the center vertex in the middle of the original top half of the cylinder, use circle select to get it, otherwise you will be RMB a number of times to find it, and move G it up along the Z axis.

Select all A , and then W and Remove Doubles, in case any were created.

2.42.6 Extrude the parachute straps

Now go into top-view by pressing NUM7 . Here you will select ~4 edges at opposite sides of each other.

Go back into side-view NUM3 and extrude downward by pressing E and then Z .

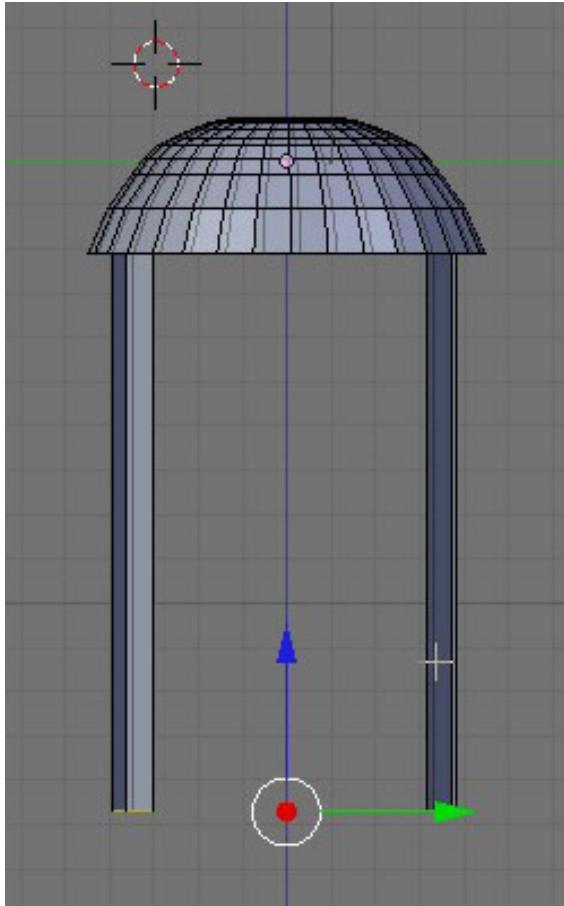


vertices.

A alternate way to make the parachute top is to create a UV sphere, and cut it in half.

2.42.7 Merge it together

All that remains to finish your parachute is to press ALT + M then choose "At Center" which will merge all selected



2.43 Model a Low Poly Head

2.43.1 Overview

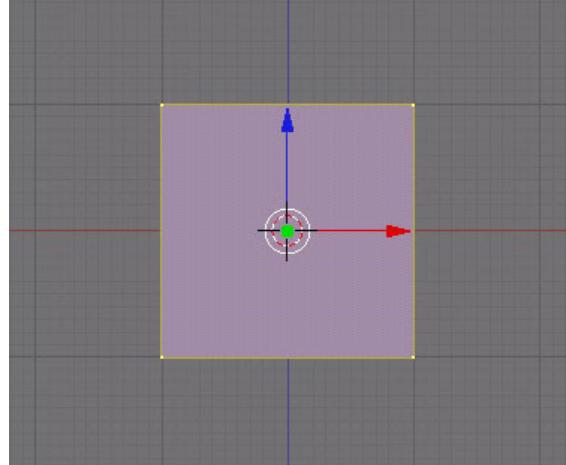
This tutorial is designed to teach users to make a low-poly animesque head in Blender.

What you need to know:

- Basic Blender controls

2.43.2 Add a plane

Noob Note: Using triangles on a subsurfed model may result in “peaks” appearing in some areas. To reduce this problem, merge as many triangles into quads as possible.



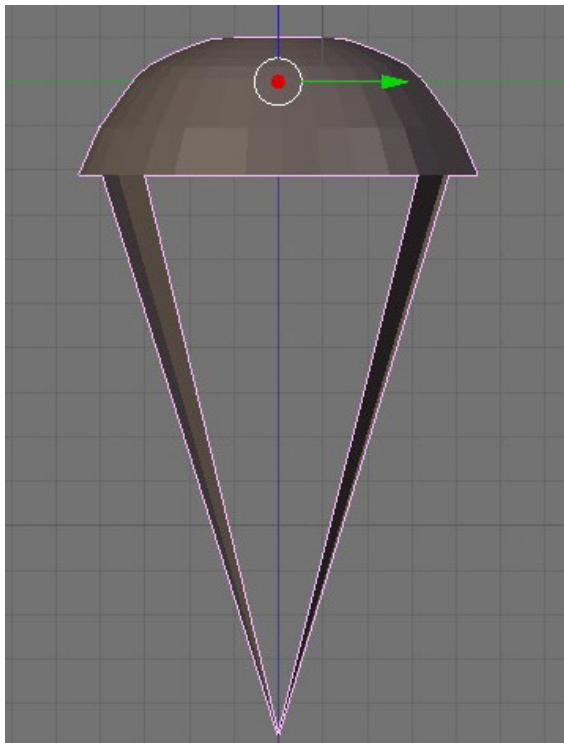
Add a plane

Start with a new file, and delete (X) the default cube.

Be in **Front** view (NUM1)

Add a plane (SHIFT + A -> add-> mesh-> plane)

Click on “Align to view” in the settings at the bottom left side.



2.43.3 Make a pointed chin

Switch to **Edit mode**

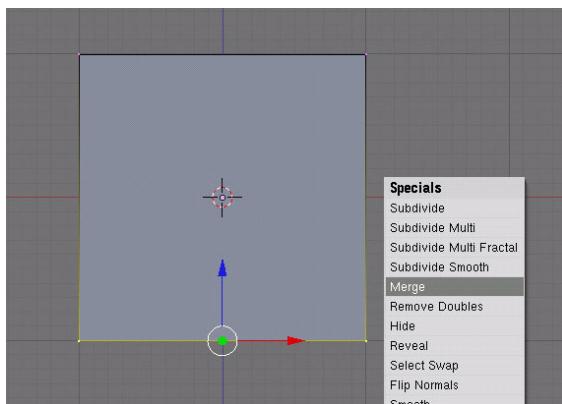
Be in **Vertex select** mode

Select the bottom two vertices (SHIFT + RMB) and press W to bring up the vertex menu. “Select Merge”, then “At Center”, or just hit ALT + M and choose “At Center”.

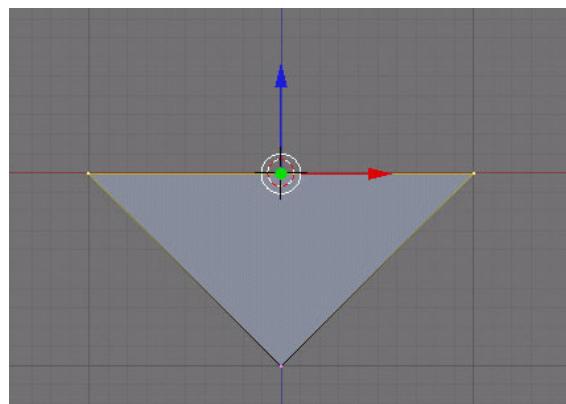
Now you have a pointed chin.

Select the top two vertices (use the B)

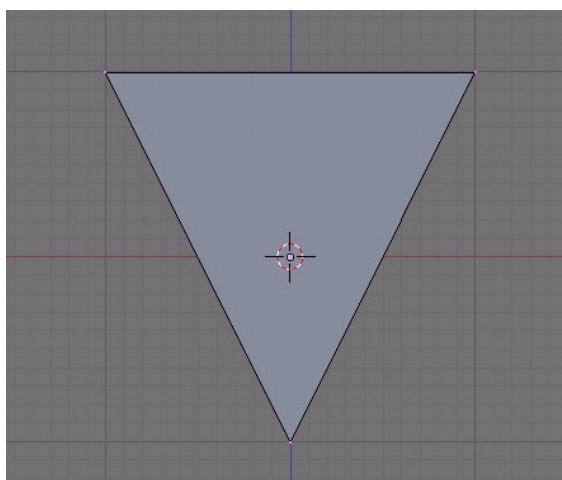
Re-arrange them so they make more of a chin shape by pressing the G to move and Z to constrain the movement



Merge the vertices

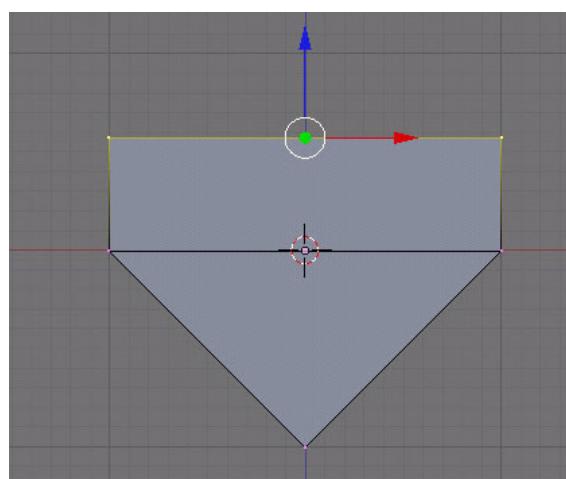


Select the top 2 vertices

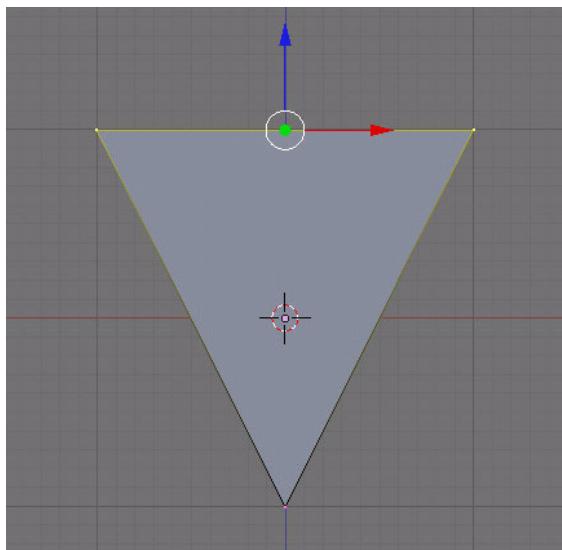


A pointed chin!

2.43.4 Extrude the face

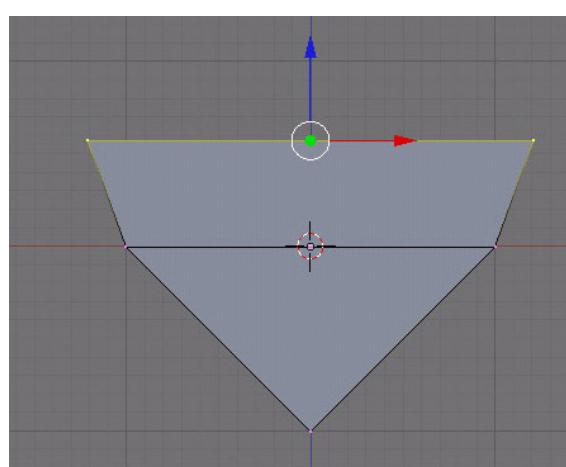


Extrude along the Z-Axis



A pointed chin!

Now extrude (E) “edges only”, along the Z-Axis (Z) so that you have another area.



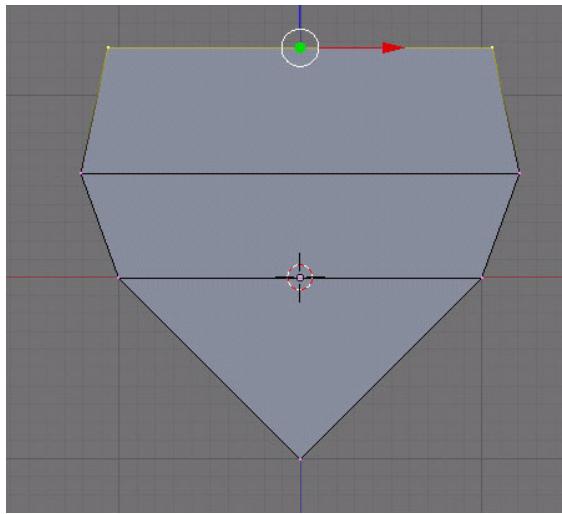
Scale it

to the Z-Axis.

Noob Note: Move it down; don't scale it out.

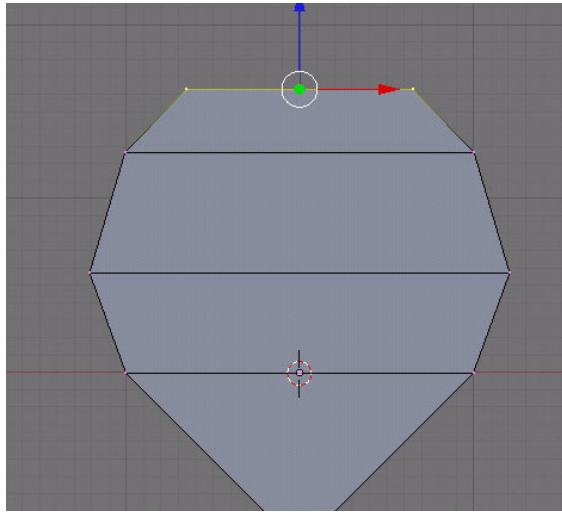
As a precaution, remove doubles. (**WKEY**)

Scale it (S) so that it's not so cubic.



Extrude along the Z-Axis again

Now extrude along the Z-Axis again (E then Z) and scale (S) it down a bit.



Extrude along the Z-Axis again

One last time... (E) (Z) (S)

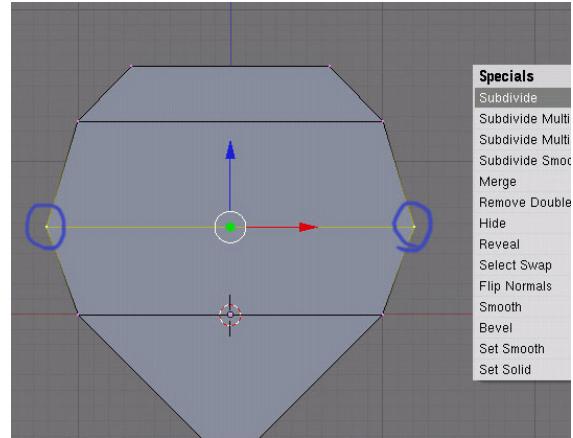
2.43.5 Make the facial features

Select the middle vertices (**BKEY**) or (**RMB**) and press (**WKEY**) to bring up the specials menu. Subdivide it once.

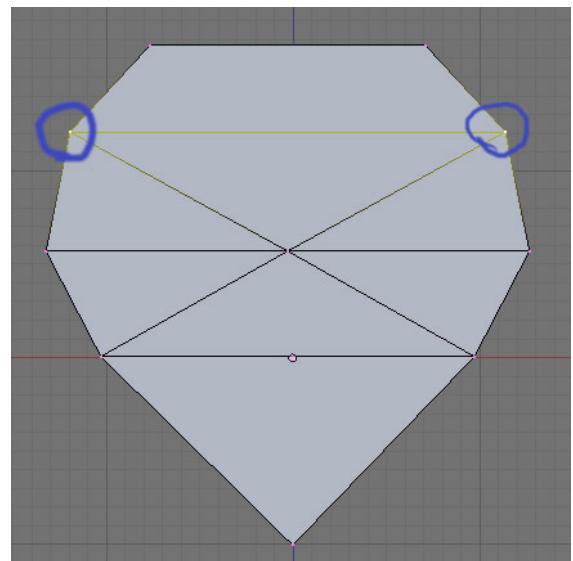
Note: In newer versions you have to check the checkbox: “Quad/Tri Mode” in the lower left corner.

Now select at first the pair of vertices above and subdivide once.

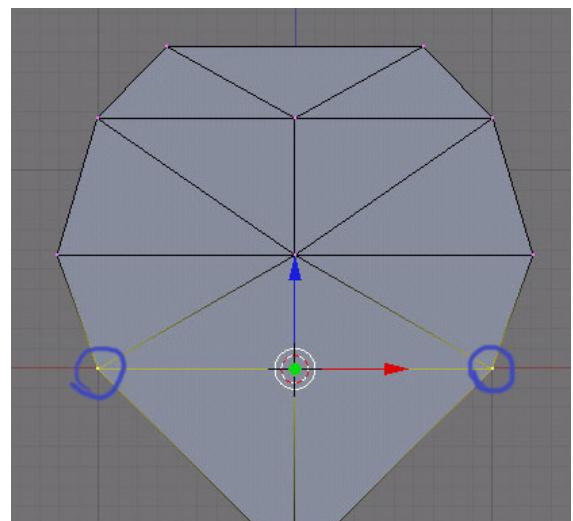
Now subdivide 2 times with the pair of vertices below the center line.



Select the middle vertices

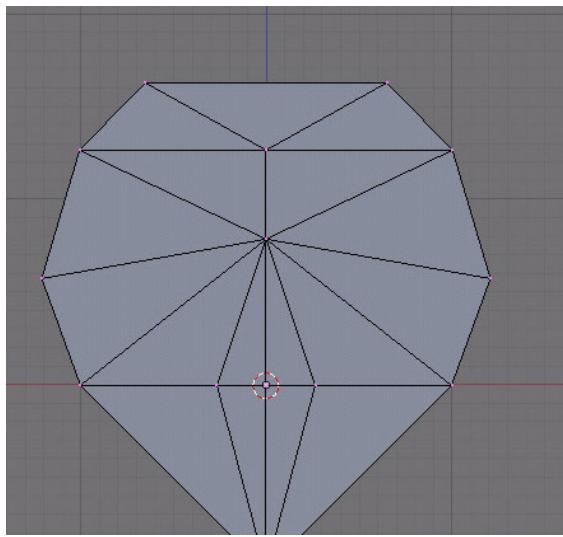


Select the vertices

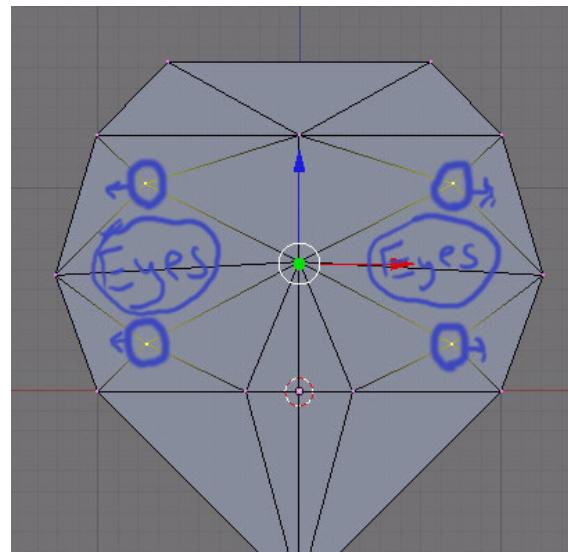


Subdivide!

Yours should now look like this one on the right.

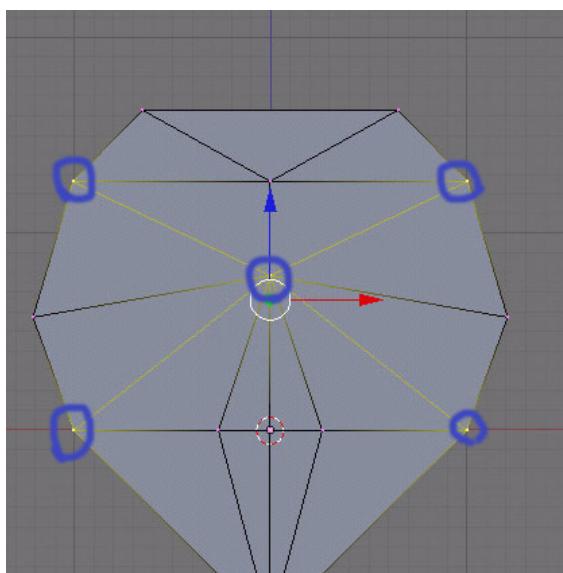


There!



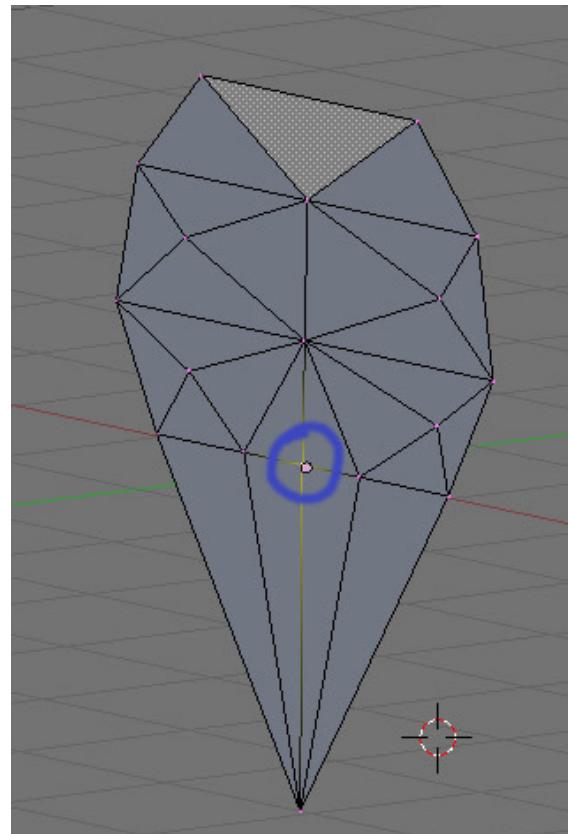
Make some eyes

Eyes



Subdivide some more

Select and subdivide here, so you can make the eyes.
And rearrange the vertices so they make eye shapes.



Select these

Nose

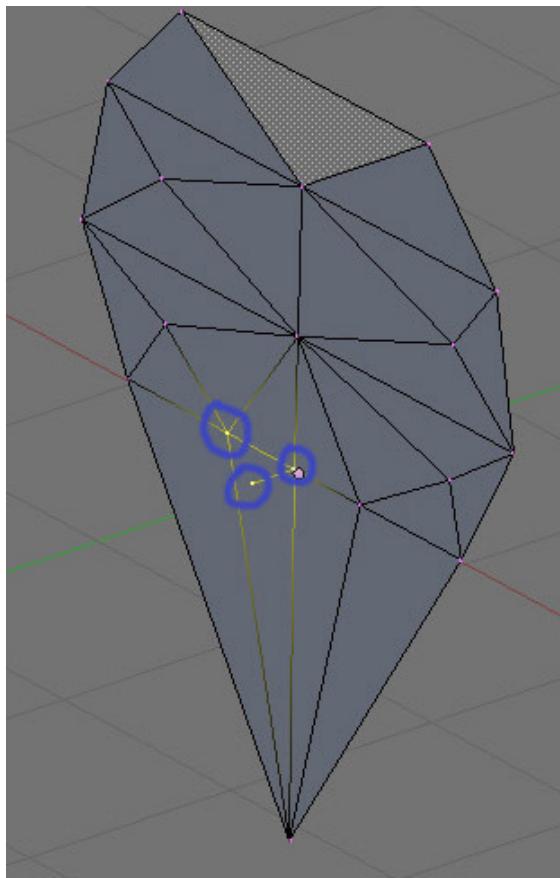
Now select the middle vertex here (right mouse button).
Extrude it along the Y-Axis (**EKEY**) (**YKEY**) and move your mouse around to change how far it moves.
Select these three vertices ('**SHIFT+RMB**') and press (**F**) to make a face.

Now do the same to the other side, and you will have the nose's base.

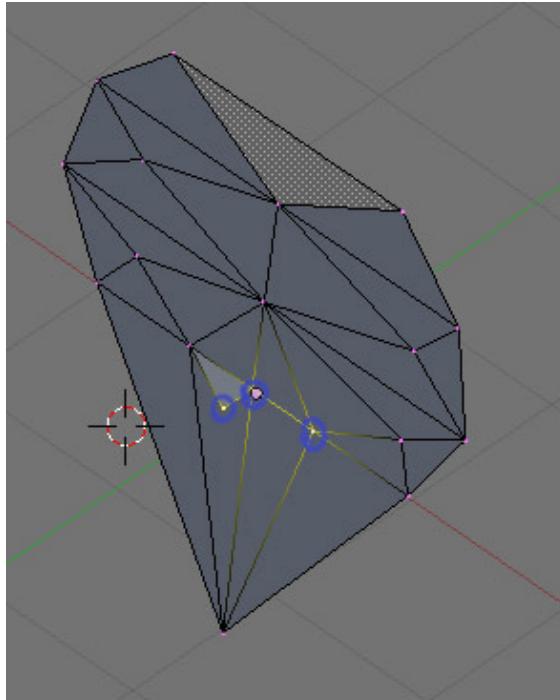
Select these vertices (**SHIFT+RMB**) and make a face (**F**).

Do this to the other side.

Now you have a nose.



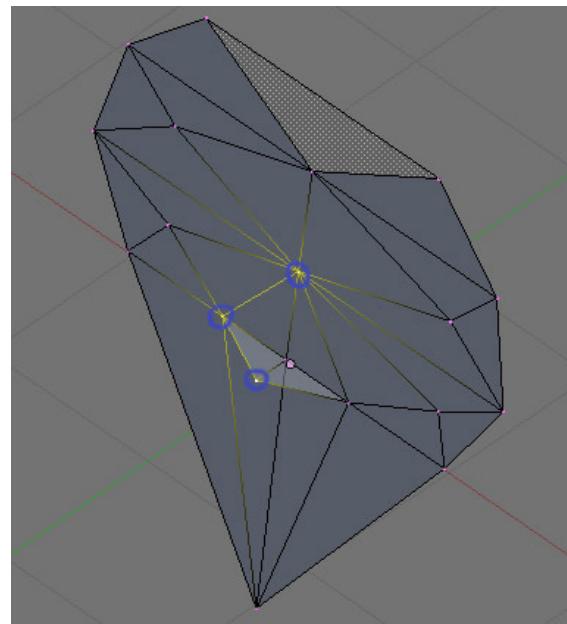
Let's make a nose!



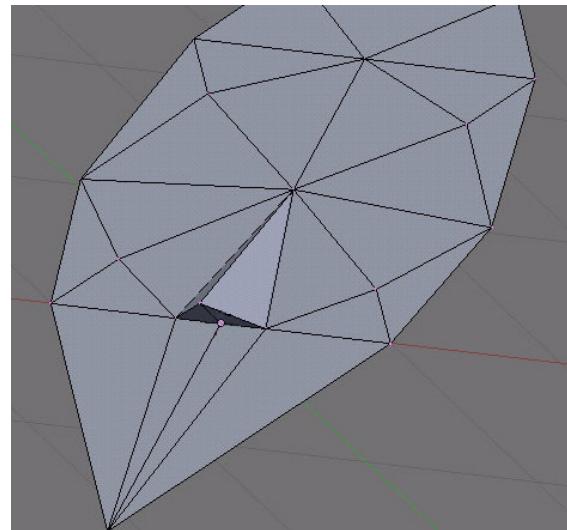
Do the same to the other side

Remove interior nose faces

Rotate the camera (**CTRL+NUM1**) so you are seeing the back.



Make a face



You have a nose!

Be in **Face select** mode.

Select (**SHIFT+RMB**) the faces behind the nose.

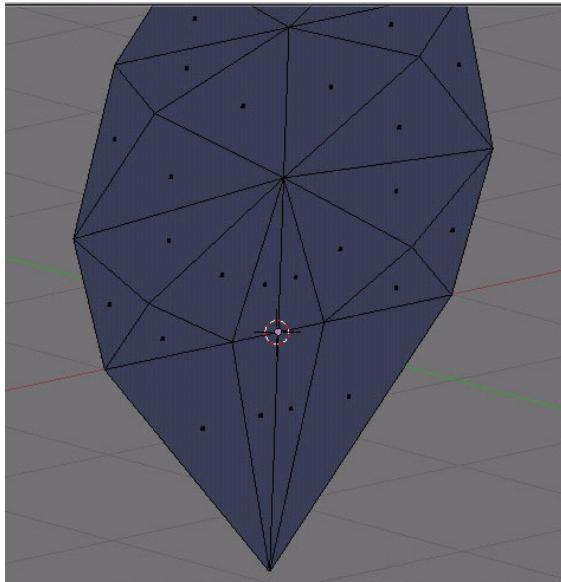
Delete (**XKEY**) them (on the popup pick “Faces”).

2.43.6 Smooth the mouth region

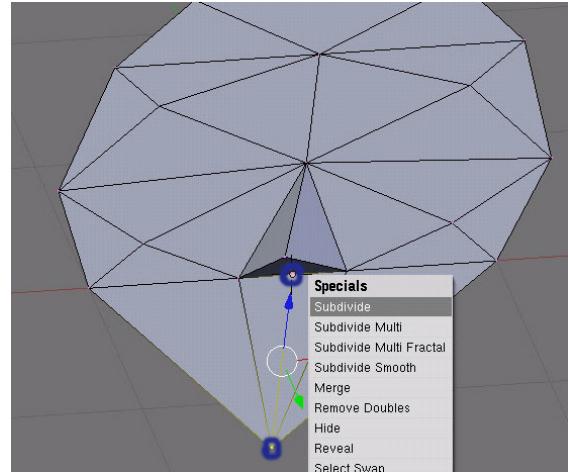
Be in **Vertex select** mode

Select the vertex at the bottom of the chin, and the one at the bottom of the nose.

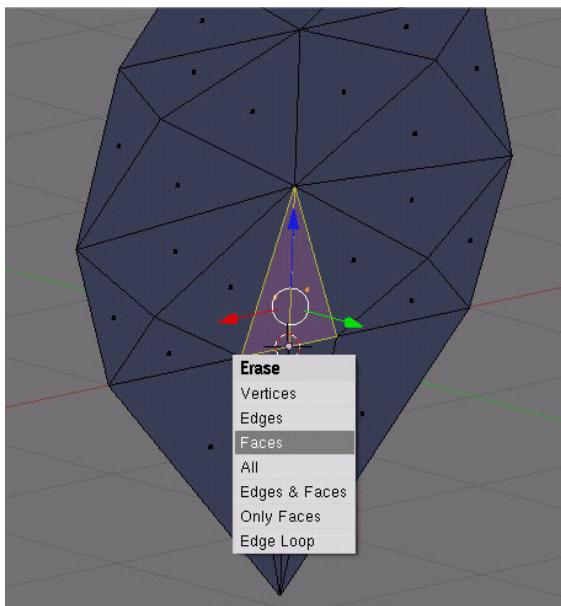
Subdivide them.



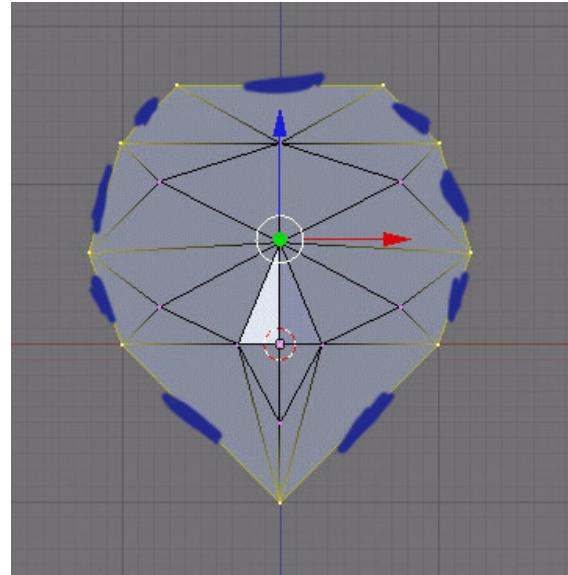
Rotate the camera



Select these



Get rid of these



Select edges on the outside

2.43.7 Make the head

Be in **Edge select** mode

Select all the edges on the outside (shown with blue lines in the photo).

Move (**GKEY**) the vertices back, along the Y axis (**YKEY**), a bit to give the face more smoothness

Be in **Vertex select** mode

Rearrange the chin, by moving (**GKEY**) the bottom vertex along the Y axis (**YKEY**) till it looks right.

2.43.8 Make the back of the head

Be in **Edge select** mode

Now select the edges all around the back of the head, using (**BKEY**).

Extrude these along the y axis (**EKEY**) (**YKEY**)

Ta da!

Now select a set of 4 back vertices that form sort of a rectangle and press (**FKEY**) to make a face.

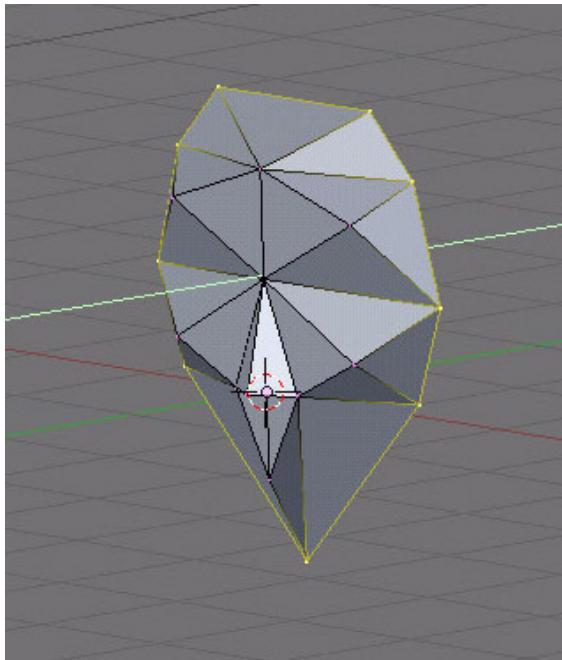
and so on....

and so on....

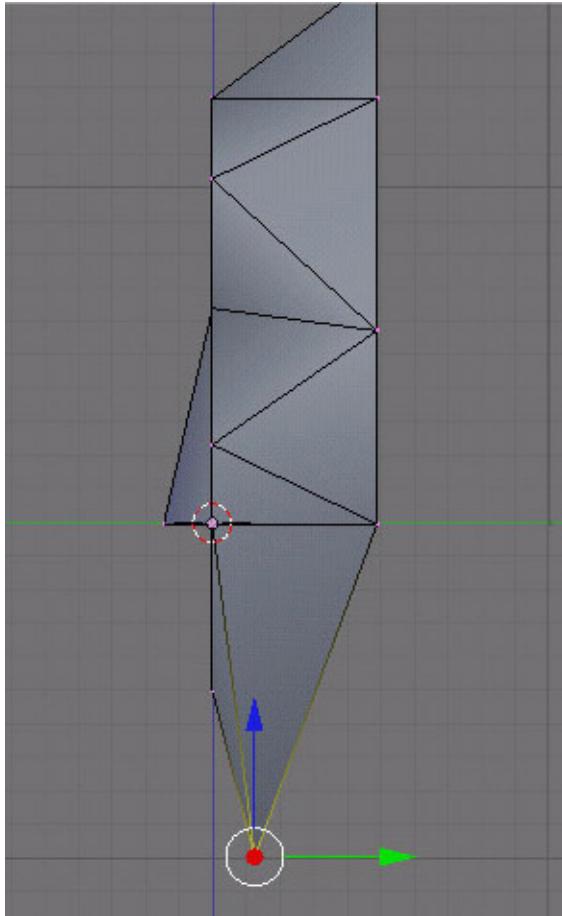
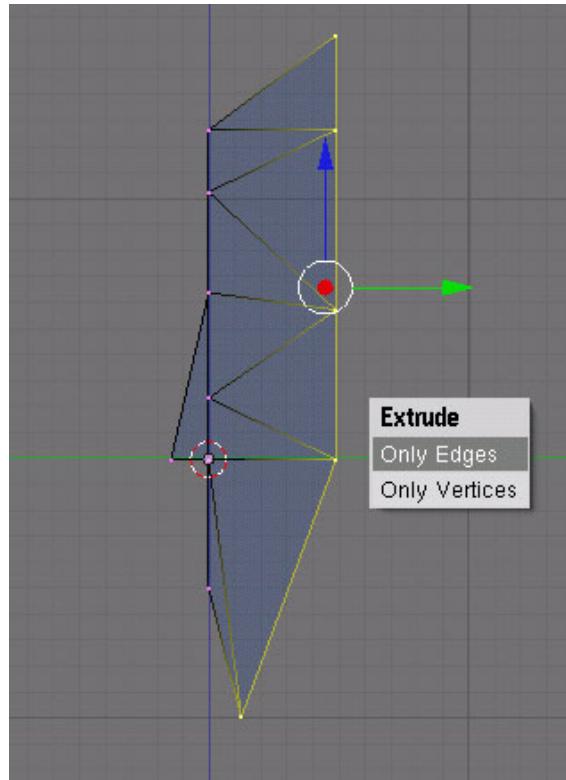
now select and subdivide these once (**W**)

and these points too.

Now select the middle and move it back (**G**) (**Y**).

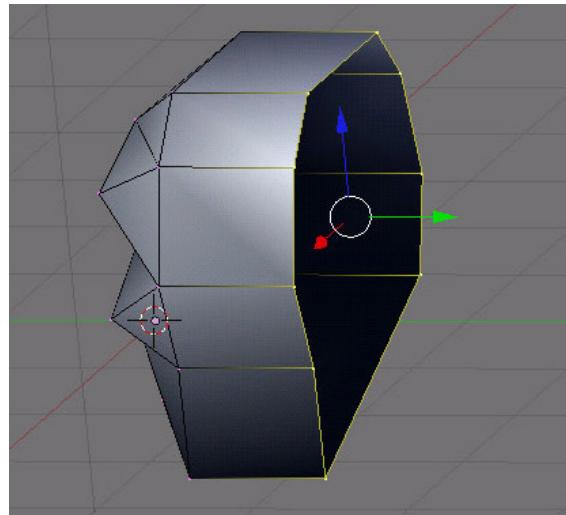


Give the face more smoothness



Select the vertices at the back of the head

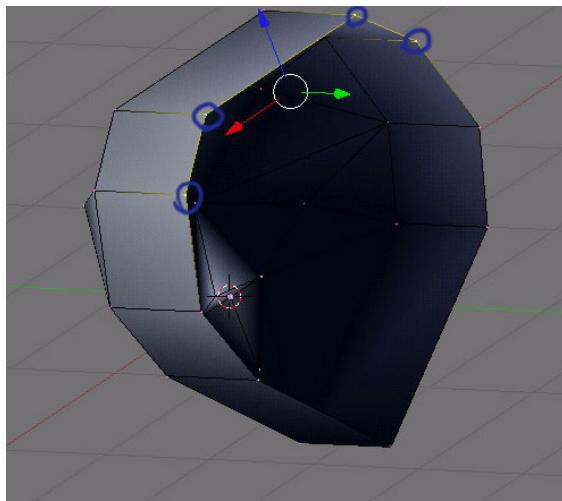
Extrude along Y-Aixis



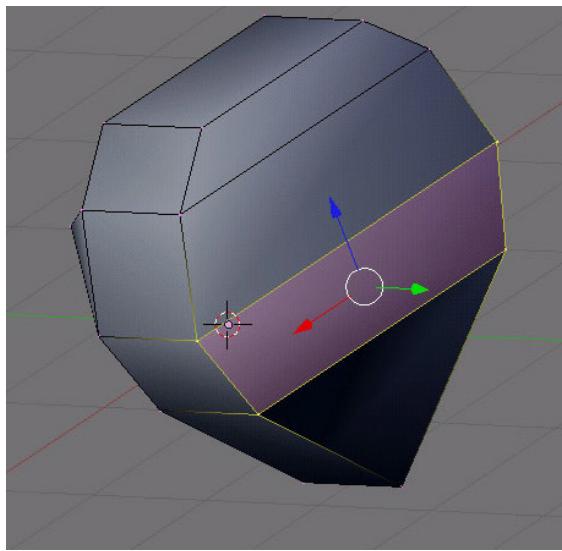
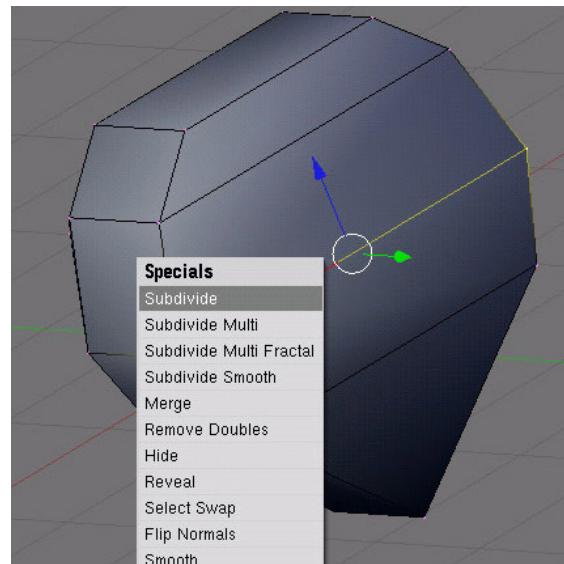
Ta da!

2.43.9 Finishing it up

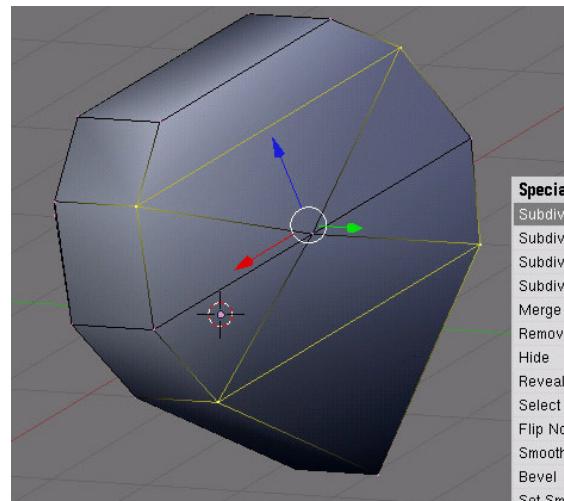
And here it is with sub-surfacing. You have finished. Hit (F12) to see the final render.



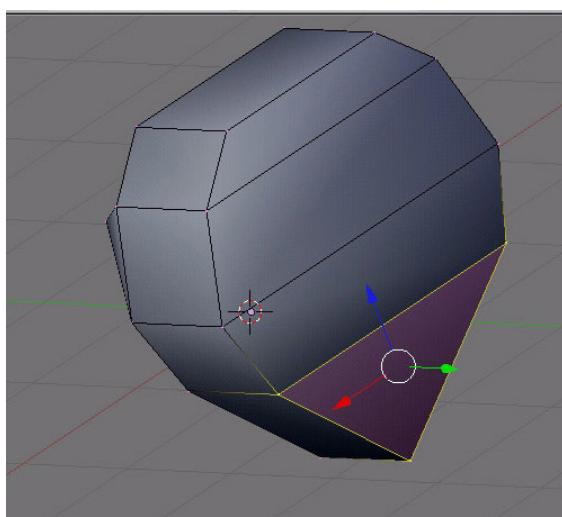
Make a face



Subdivide these

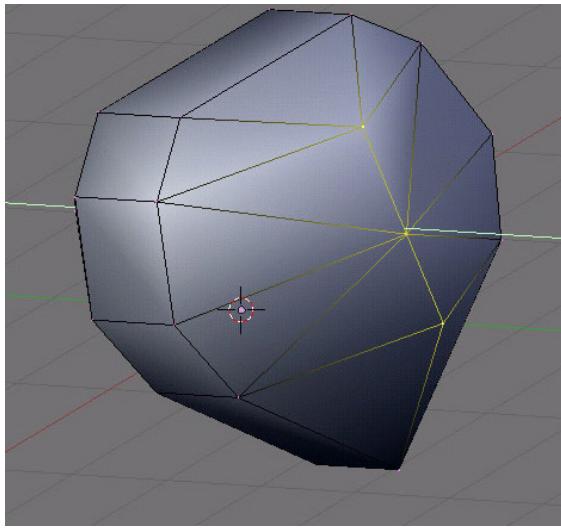


and so on...

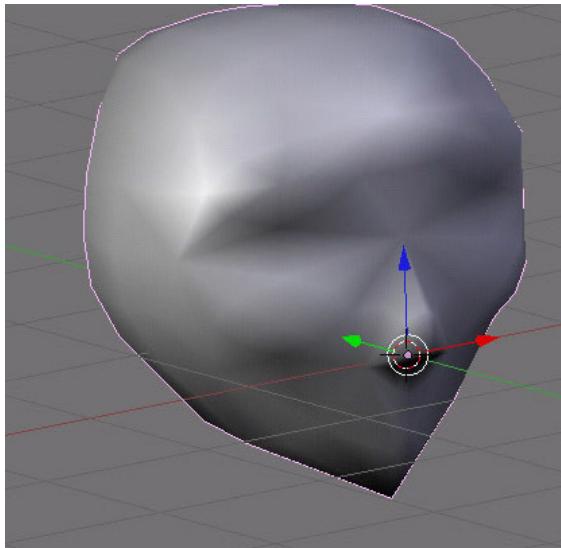


Subdivide these as well

and so on...



Move the middle back



With subsurfing

2.44 Building a House

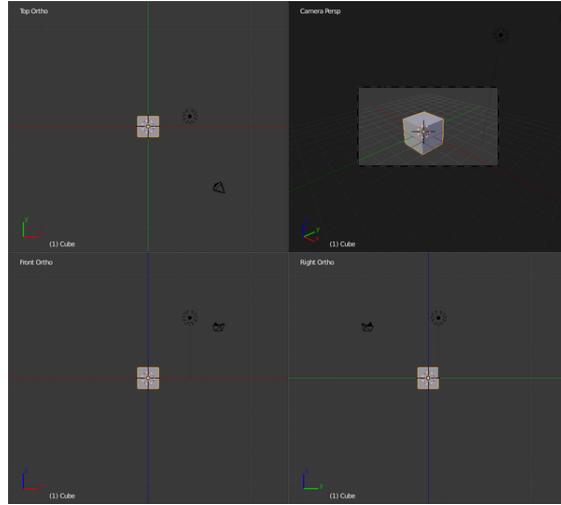
In this tutorial, you will learn how to make a simple toy-like house with a fence around it. You will learn how to use *array modifiers* to replicate the palings of the fence, saving you from duplicating them one by one.

This tutorial is based on Bart Veldhuizen's "Building a House" from Tutorial #01 published by NaN in 1999 which is also available in a PDF at <http://download.blender.org/documentation/BlenderTutorialGuide1.tar.gz> (a tar.gz containing BlenderTutorialGuide1.pdf) or <http://download.blender.org/documentation/BlenderTutorialGuide1.zip> (a ZIP file containing the same PDF).

Permission was asked to use it and Ton Roosendaal said "Be assured that everything that was produced by NaN now is open and free content for everyone to reuse, in-

cluding the tutorial "Building a House"."

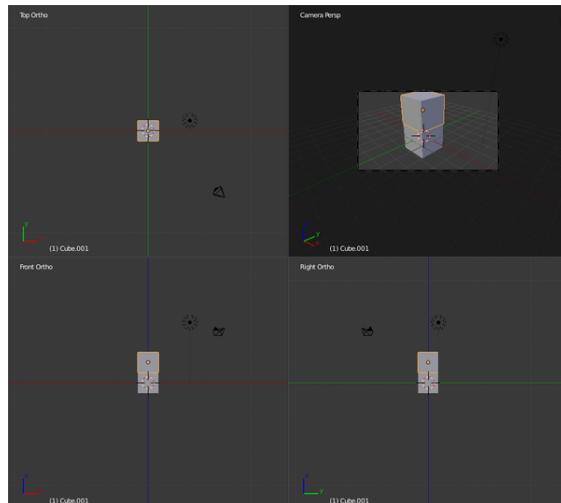
2.44.1 Setting the Scene



Start a new document (CTRL + N). Split the view into four views as at right, by pressing ALT + CTRL + Q . This gives you a standard set of views: top, front, side and camera, the first three orthographic and the last one in perspective. You can switch back to the single 3D view at any time by pressing ALT + CTRL + Q again.

Leave the default cube, it will be the walls of the house.

2.44.2 Make the Roof



Ensure you are in **object** mode (TAB switches between object and edit mode)

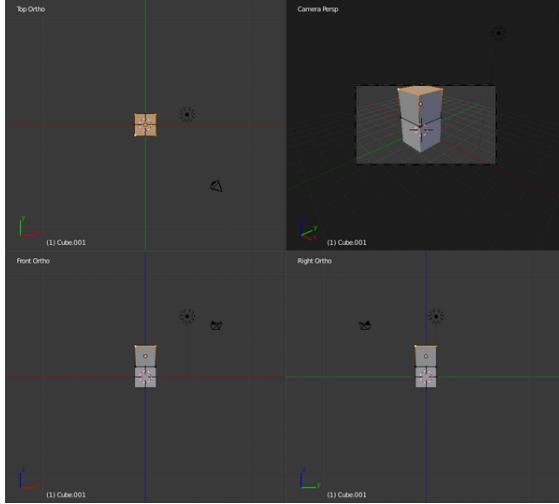
Now, in the Front view window (lower left)

Press SHIFT + D to duplicate the cube, grab mode is automatically selected, press Z to restrict the move to the

up-down axis and move the cube to rest on top of the original (pressing CTRL while moving will snap it to the grid and make it easier to position accurately.) Press ENTER when it is in place.

You can see how useful this four-paned window is; it shows you exactly what's going on.

Note: It helps to simply grab one of the arrows, press Z and then press 2 on the key pad, this will move the block 2 units in the z direction. the standard starting cube is 2 units tall.



The top cube is going to become the roof and needs to be given a triangular cross-section.

Select the top cube (if it is not already selected) by clicking RMB on it

Press TAB to go into **Edit Mode** (check the box in the middle of the 3D window header)

Press A to deselect all the vertices.

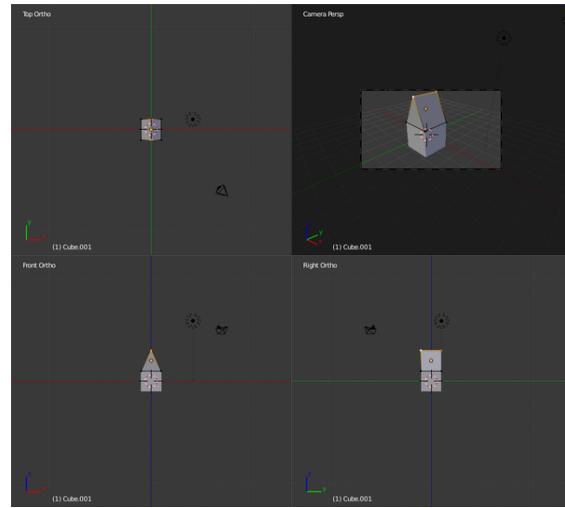
In the top view, select the top four (RMB the first one and SHIFT + RMB the rest). You can check in the other views that only those four are selected.

Note: another way to do this is to have “limit selection to visible” turned off, and then press the B for, border select, and make a box around the top vertices. (it helps to view it from one of the sides, by pressing either the 1 or 3 keys)

Another Note: Or, with ‘limit selection to visible’ turned on, press C for circle select and turn the mouse key to make the circle large enough to enclose all four vertices before you press LMB .

Make the Apex of the Roof

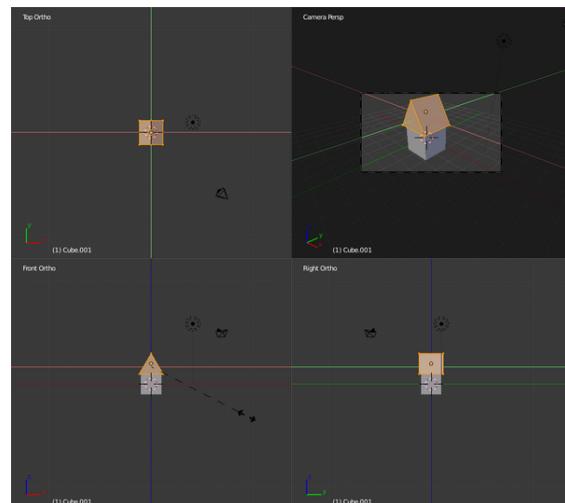
Now press S for scaling and X to limit the scaling to the X-axis (left to right). You could try to move the mouse back and forth to bring the top vertices together, but you will have a hard time lining them up exactly, so it's best to just type in 0KEY (zero) followed by ENTER to set



scaling to zero.

Now it looks like there are only 2 vertices where there were 4, but in fact there are still 4, even though 2 of them are occupying exactly the same positions as the other 2. With these 4 still selected, bring up the vertex specials menu with W and select “Remove Doubles”. You should see a message briefly flash up at the top right saying “Removed 2 vertices”. Now there really are only 2 vertices where there previously were 4.

Form the Eaves



The roof needs to project over the walls of the house to form the eaves. To do this, we will scale it, but only along the X- and Y-axes, not the Z, so it doesn't become taller.

Select all the vertices in the roof object by pressing A once or twice.

Press S to scale, followed by SHIFT + Z to scale uniformly along all axes *except* Z, and scale to about 1.1. Confirm the operation with LMB or ENTER in the usual

way.

2.44.3 Naming the Roof and House Objects

In a complex project with lots of objects, it can be helpful to keep them straight by giving them names. This is less of an issue in a simple tutorial like this one, but for practice, let's give names to your objects anyway.

Go to the Object context  in the Properties window, and at the top you should see an editable field containing the name of the currently-selected object. The walls of the house should be called "Cube", and the roof should be called "Cube.001" (note the automatic addition of a numeric suffix to keep the names unique). Try changing these to, say, "House" and "Roof" respectively.

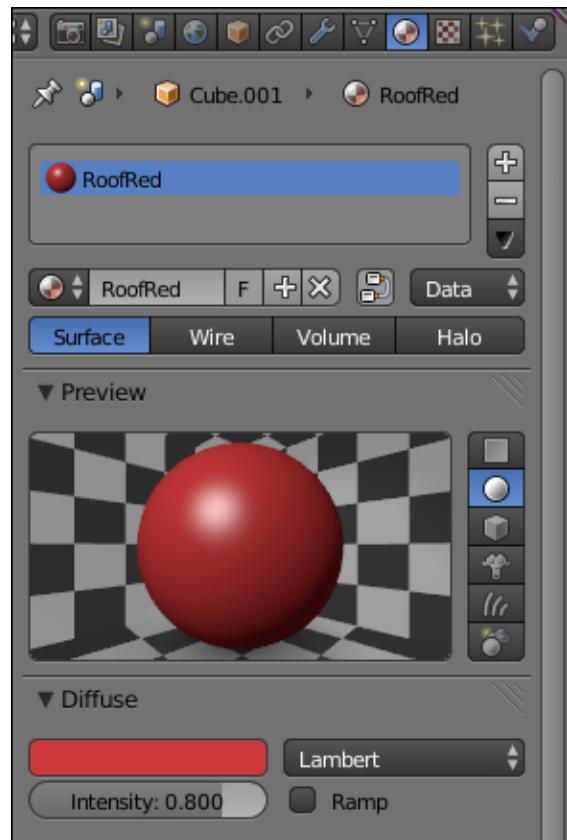
Object Name Versus Mesh Name

If you look in the Mesh Data context , you will also see a name like "Cube" or "Cube.001". This is a separate name for the object-type-specific data (the mesh data, in this case), quite independent of the object name. Don't worry about this name for now; you will learn about its significance later.

2.44.4 Colouring the House

The roof now needs to be given a different colour to the default grey. Select the Materials Context  in the Object Properties window. In the list at the top, you should see a single entry, called "Material" (the name of the initial default material). Below that is an editable field containing the name, and immediately to its right should be a small box with the number "2" in it. This number indicates that the same material is being used in two places—in this case, we know the other place is the object representing the walls of the house. Click on this "2", and that will force a new copy of the material to be made (leaving the house walls with the original); the number will disappear, and the material name will change to "Material.001" to be different from the original.

Change this material name to "RoofRed", to make it clearer what it is for. Next, find the panel "Diffuse" further down, showing a swatch of the diffuse (non-reflective) colour, which is initially white. Click on this to bring up a colour picker, and choose some suitable shade of red for the roof. (Notice that when using shades of red, the white to black slider on the right actually creates shades of brown from the red.) As you do this, the 3D view should instantly update to show your new colour being applied to the roof of the house. If you want to exit the colour picker and leave the colour unchanged, press



ESC ; otherwise, to confirm your choice, simply move the mouse outside the picker window, and it will close, leaving your last-chosen colour in effect.

Follow a similar procedure to choose the colour for the house walls: click the "House" object with RMB in one of the 3D views, go to the Materials Context  in Object Properties, change the material name from "Material" to something more appropriate (here I'm choosing "HouseSandy" because I'm going to give the house walls a sandy-yellow colour), click on the colour swatch in the "Diffuse" panel and choose a suitable colour for this material. **Noob Note:** I have found it very useful, when I re-name anything, to type my new name in ALL CAPS. This makes it easier for me to pick out what I have done. Your house should look like this now.

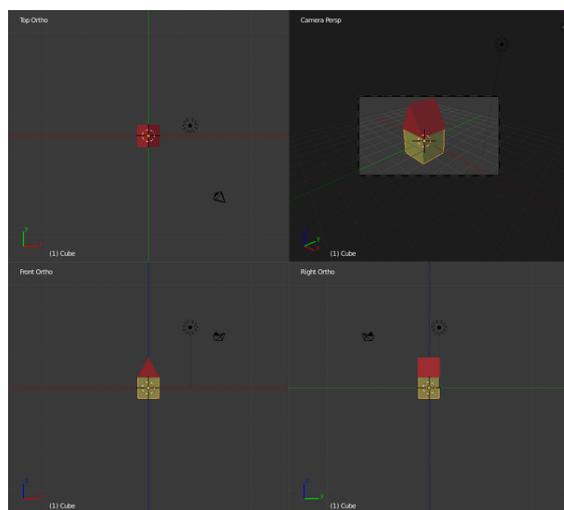
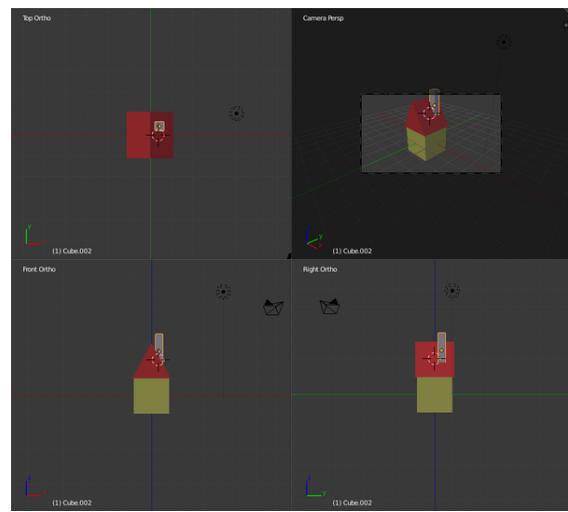
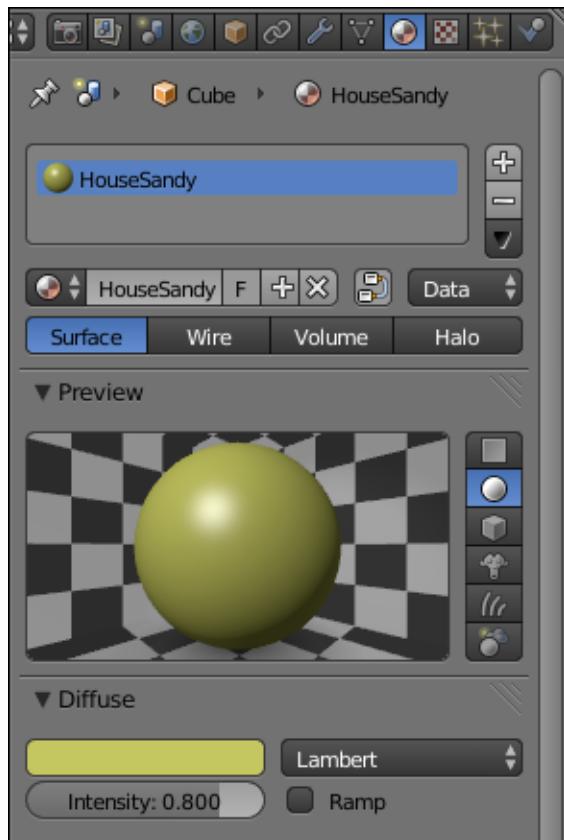
2.44.5 Make a Chimney

Go into Object mode, if you're not already in it. (TAB flips between object and edit mode.)

Go to the Front view (lower left window).

Click LMB to position the 3D cursor on the right-hand side of the roof, this will select the spot where the chimney will be created.

Press SHIFT + A to Add → Mesh → Cube from the popup menu — a bit large, isn't it?



Name and Colour the Chimney

In the Object context in the Object Properties window, change the name of the chimney object to “Chimney”.

Next in the Materials Context in the Object Properties window, you will notice there are no buttons, and there is an icon , and next to that a button with the word “New” in it. You can now choose to have the chimney a different colour to the roof or the house or reuse one of these colours.

If you want a new colour, the “New” button, and it will create a new material with a default name and settings. You can rename this and give it an appropriate colour.

If you want to reuse the same colour, you can click on the icon and select from the available materials, namely “RoofRed” and “HouseSandy”.

2.44.6 Adding a Window

Make the Window Frame

Be in Object mode.

Using the front and top views, click LMB to place the 3D cursor where you want the window to go: use the top view to place it against the front wall, and the front view to place it a little to the left and up from the centre of the wall. (Of course, it’s easy enough to reposition the window after you’ve created it, so exact initial placement is not critical.)

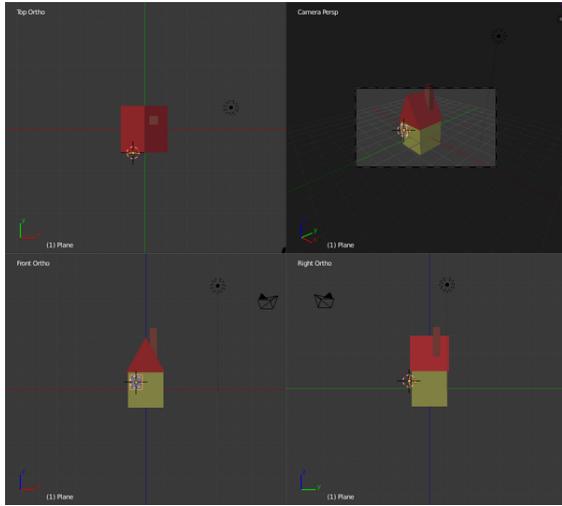
In the front view (lower left) add a plane: SHIFT + A Mesh → Plane). This will initially be laying flat, facing upwards instead of forwards; to fix this, just rotate it 90° about the X-axis: R X 9KEY 0KEY ENTER .

Scale it along the X and Y (S , SHIFT + Z) to about 0.2.

Now grab it (G , X) and move it into position looking at the front view (lower left window). (Feel free to use the mouse wheel to zoom in the view to make things easier to see.)

When this is correct, move it into position along the Y axis (G , Y) looking at the top view (upper left window).

You can check the side view (lower right window) and the camera view (upper right window) to see that it looks OK.



Scale S it down to 0.4, and press ENTER

To give it a rectangular shape, scale the plane by about 0.8 along the X-axis S X 0KEY .KEY 8KEY .

Move it into position (G), using the top and front view, on the left hand side of the house a little higher than midway.

Be in the Side View window (lower right).

Move the plane along the Y-axis (G , Y) until it is just in front of the house. You may want to zoom in (MW) to see better.

Now, still in Side View,

Switch to **Edit** mode (TAB)

Extrude the plane towards the house (E , -0.07) until it is embedded in the wall.

Make the Window Sill

To make this box into a window frame will require a larger view: move the mouse into the Side View window and zoom in with the mouse wheel.

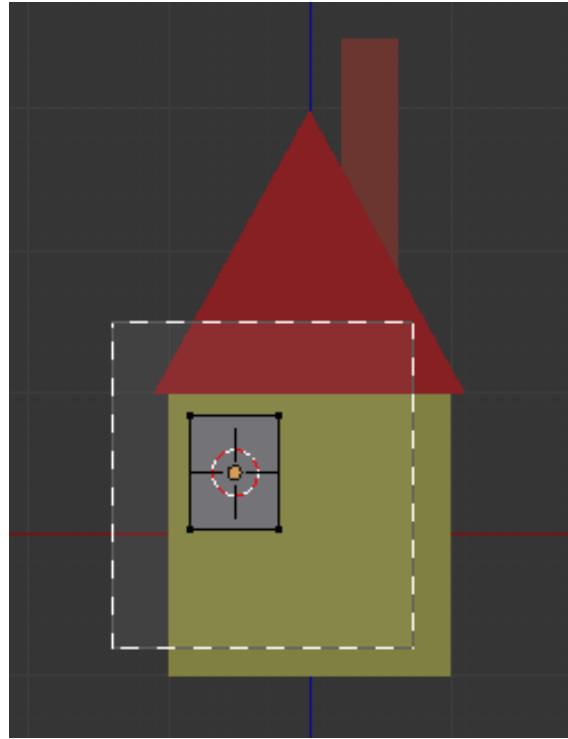
Deselect all (A) and select the outer (leftmost in the right view) vertices using box select (B).

Press E and immediately (without moving the mouse) press ESC . This will extrude, but leave the newly-created vertices on top of the previous ones.

Scale the newly created vertices by 0.9 (S 0KEY .KEY 9KEY ENTER).

Extrude again inwards to most of the depth of the box about -.03 to -.05 along the Y axis, depending on how you have your window into the wall(E , Y , -.03, ENTER).

This screenshot shows a side view in wireframe mode (use Z to switch wireframe mode on and off).



Box-select in progress

Name and Colour the Window Frame Object

Change the name of the window object, as you previously did for the rest of the house pieces. It will be initially called “Plane” (because that is the kind of mesh object you started with); change it to “WindowFrame”. Also give it a new material, and make it white. Change the material name to “WindowFrame”.

Make the Window Glass

The glass of the window has to have a different material from the frame. There are two ways to achieve this: make them separate objects, or make them the same object, with different materials assigned to different faces. We will do the latter.

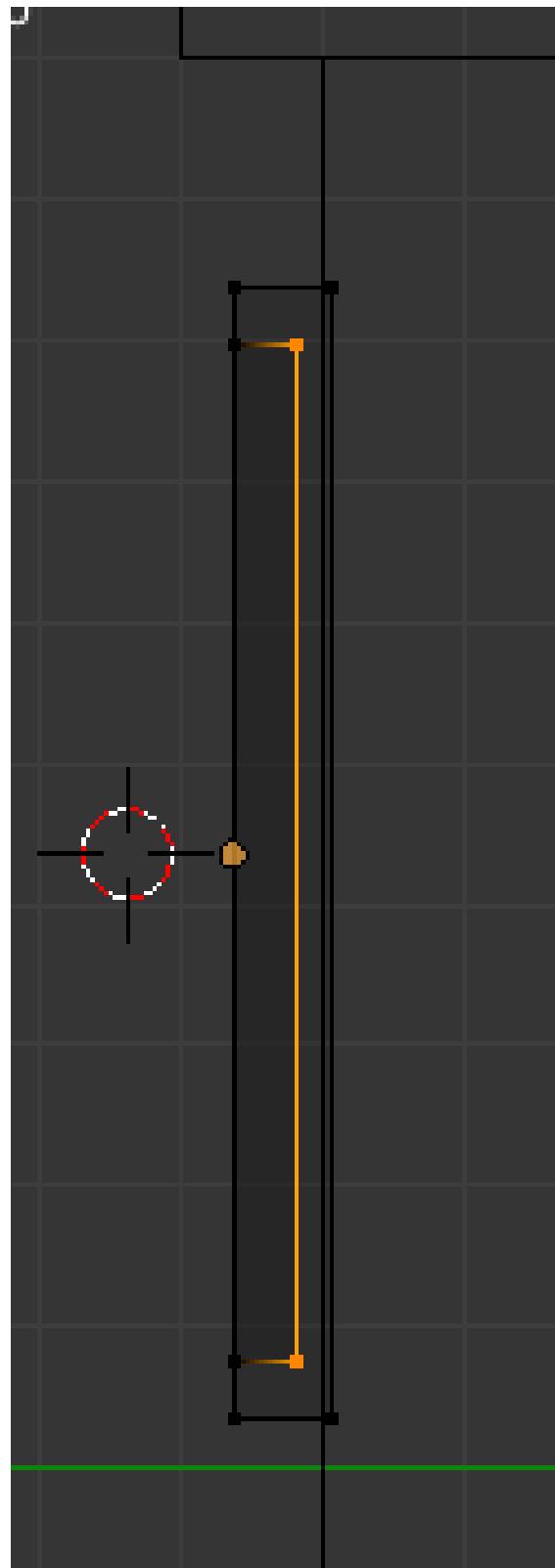
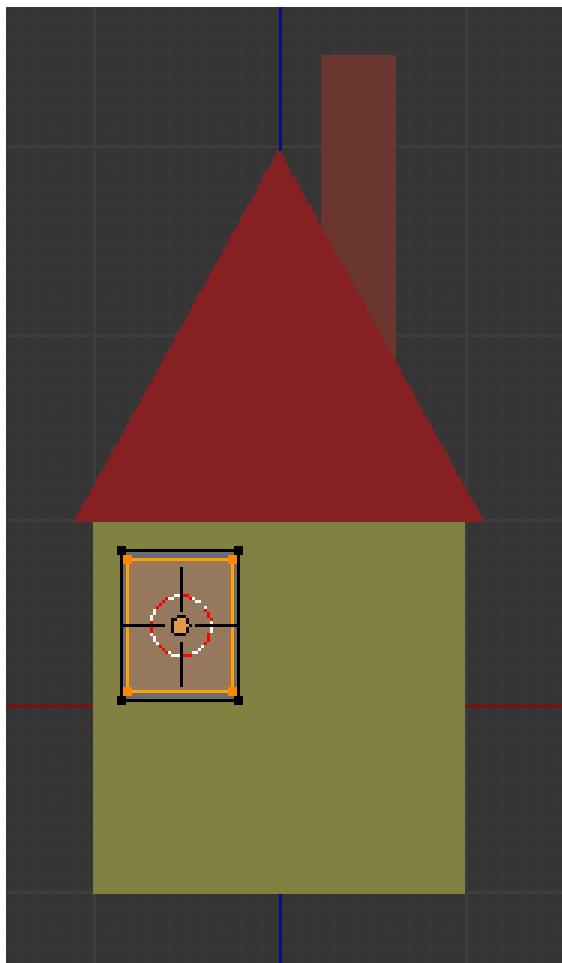
Deselect all (A)

Be in **Edit** mode .

Switch into **Face** select mode (CTRL + TAB -> faces) select mode (you've been in **Vertex** select mode up to now)

RMB on the face in the center of the window frame, to select it.

In the Materials Context, click the "+" button next to the list of materials, and where that was previously showing just the “WindowFrame” material, you should see a new blank item appear. The rest of the window will go blank, but the “New” button will appear; click this to create a new material. Give it a colour reminiscent of glass; I



chose to colour it light blue (“87ceeb” SkyBlue).

Now to connect the new material to just that one face, press the Assign button just under the material list.

Note: If you do decide to make the windows with real glass material, as explained in previous lessons, make sure you use the materials on both faces in the front and back of the window. Also use loop cuts or knife to cut out a hole in the wall in place of the window. And add a point light inside the house.

2.44.7 Adding a Door

Add the door using the same method:

Be in Object mode

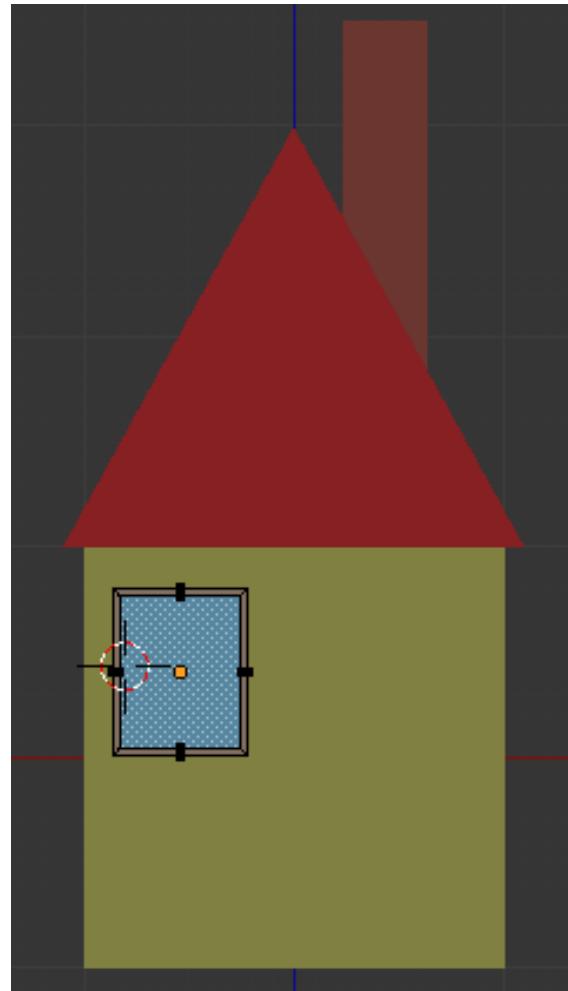
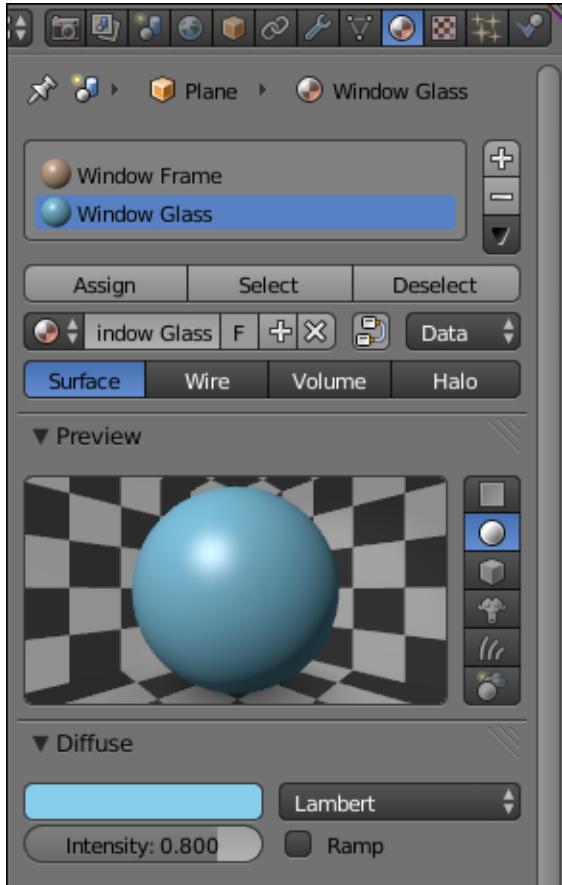
Deselect all,

Create a plane, rotating it to face forward as necessary (R X 9KEY 0KEY ENTER) to align it with the front wall of the house.

Scale the plane down by .7 and then again by .5 along the X axis.

Move it into position using Front view and then move it against the wall of the house in Side view.

Change the Object name to “Door”. Create a new material for it; change the material’s name to “Door” and choose an appropriate colour for it.



2.44.8 Building the Fence

The fence will consist of long sequences of identical palings. Creating them one by one would be a long, tedious process—which is something the computer, not a human, should do, right? In fact, even creating just *one* paling and making duplicate copies would be a long, tedious process. Which is where the magic of modifiers comes in.

Blender's *array modifier* lets you make any number of copies of a single object, all neatly arranged in a row or in various other ways. Being a modifier, the copying only happens at render time, so there is still only a single object to edit while modelling: change that, and the change is automatically propagated to all the copies.

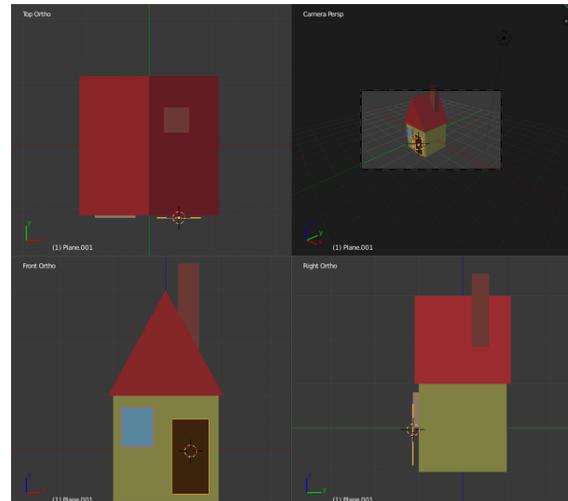
Make a Paling

Be in object mode, with nothing else selected.

Add a plane, orient it vertically (R X 90 ENTER), scaled to 0.4 and then 0.1 along the X axis in Front view.

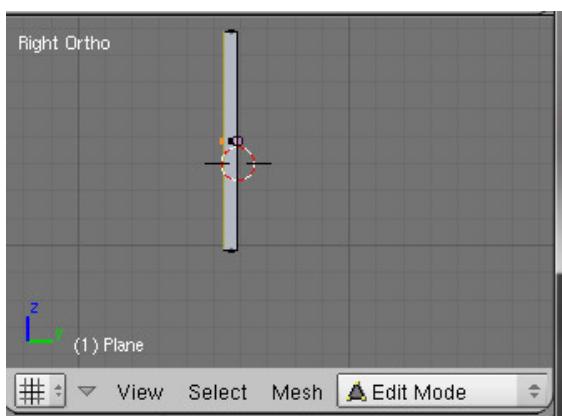
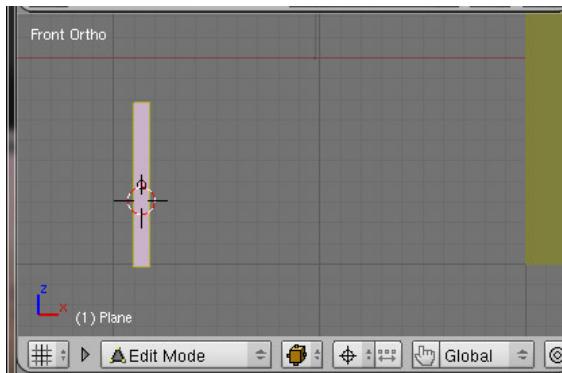
Switch to edit mode. Ensure you are in **vertex** select mode (CTRL + TAB -> "vertex"), in case you were still in face-select mode from making the window (above).

To give it some thickness, go into Side view and extrude it to 0.05.



The pointed top is made by going back into Front view, Selecting the top vertices and extruding them by 0.07 (along the Z axis is selected automatically),

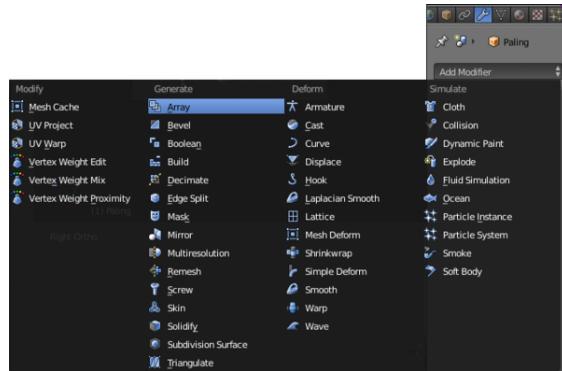
Scaling them to 0 along the X axis and then removing the doubled vertices (W , "Remove Doubles": it should say "Removed 2 vertices").



In Front view again, check that the bottom of the paling is level with the bottom of the house.

Change the name of the object and the mesh to “Paling”, create a new material for it called “Paling” and make the colour white.

Duplicate the Paling



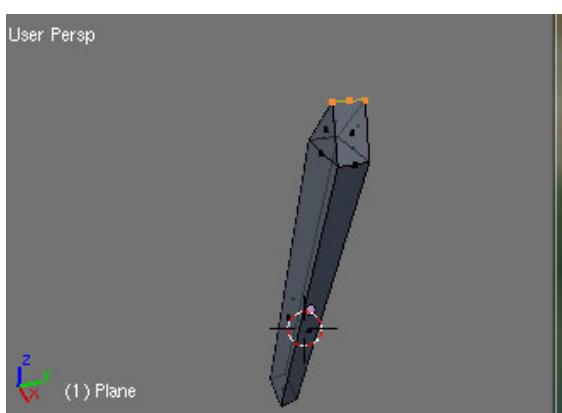
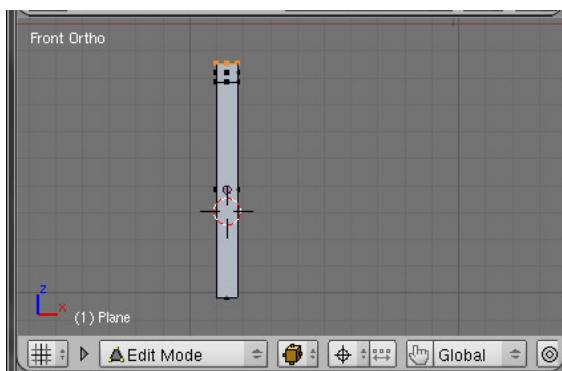
Be in Front view, as the flat face of the paling is along the X axis and it is easier to create the front fence first.

Change to Object mode. Select the paling. Find the in the Object Properties window. Click the “Add Modifier” button, and in the menu that pops up, select “Array”. You should see a second copy of the paling appear next to the first.

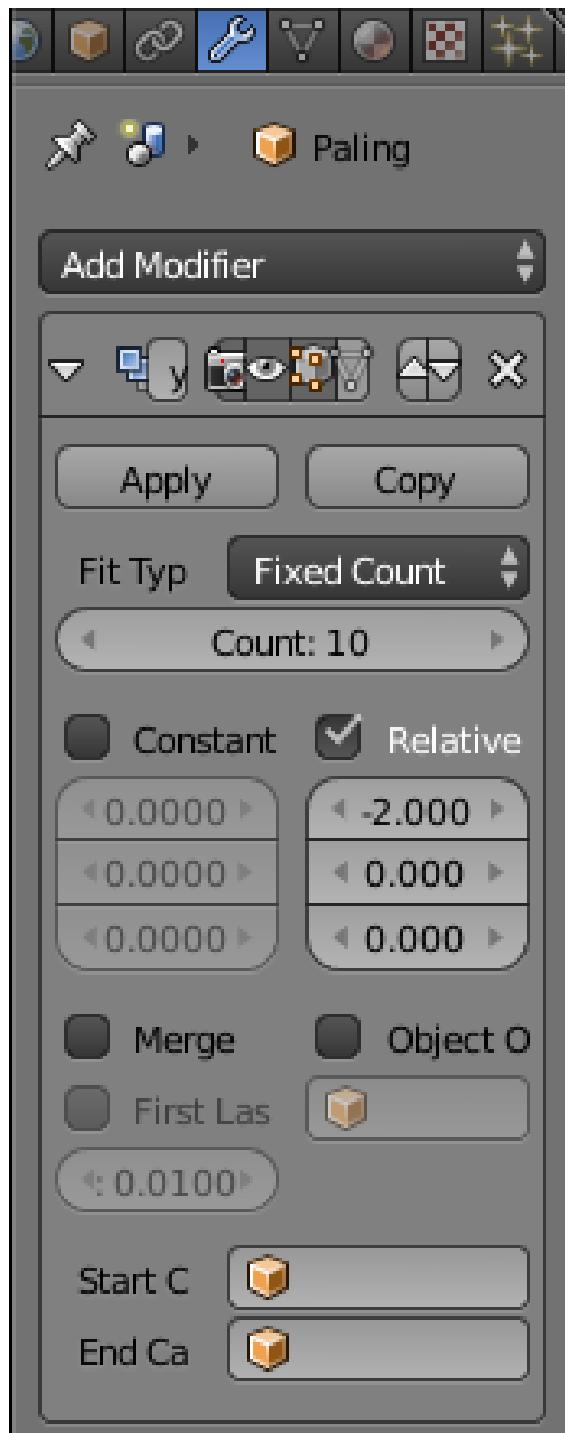
Look at the checkboxes: “Relative Offset” should be checked, “Constant Offset” and “Object Offset” should *not* be checked. Under “Relative Offset”, there are values for X, Y and Z; these are the distances between copies, relative to the dimensions of the object. Change the X value to something like -2 (the negative value causes the duplicated paling to appear on the other side, away from the gap we want to leave in front of the door). Now increase the number under the popup that says “Fixed Count”, and watch the line of duplicates lengthen until it reaches the point that you want to be the corner of the fence line—a count of about 12 did it for me.

Now type ALT + D to make a duplicate of the row of palings, followed by X to constrain its movement to the X-axis (parallel to the row of palings). Move the duplicate row to the end of the existing row so it looks like a continuation of it. Now type R Z -90 ENTER to rotate it so it lies parallel to the side of the house. Increase the array count for this row to something like 24 so it extends all the way past the side of the house to a suitable corner point.

Again, type ALT + D to duplicate this second row, this time followed by Y to constrain its movement parallel to the side of the house. Move the duplicate so its looks like a continuation of the same row. Rotate it parallel to the back of the house by typing R Z -90 ENTER . Adjust its



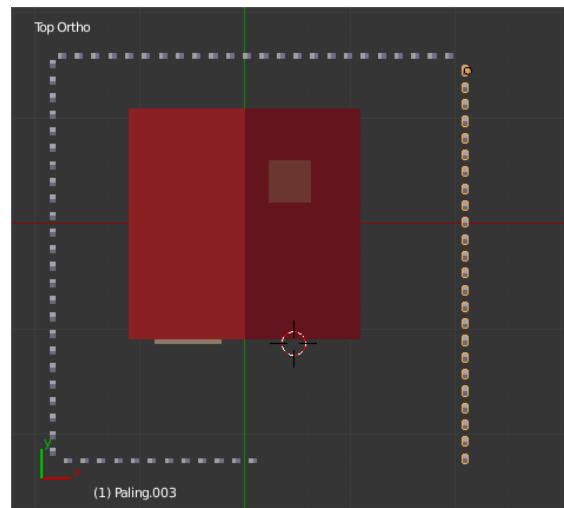
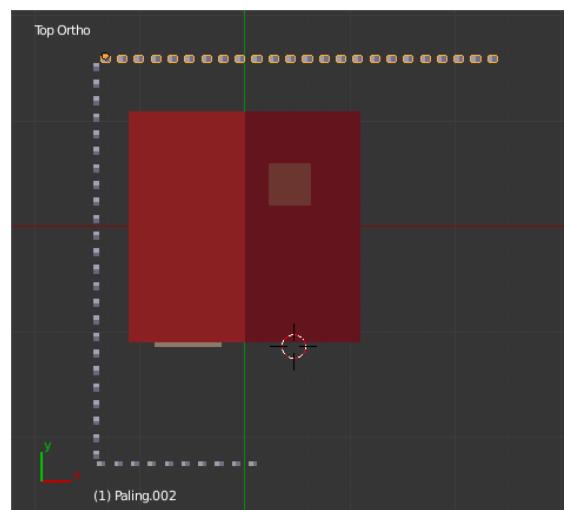
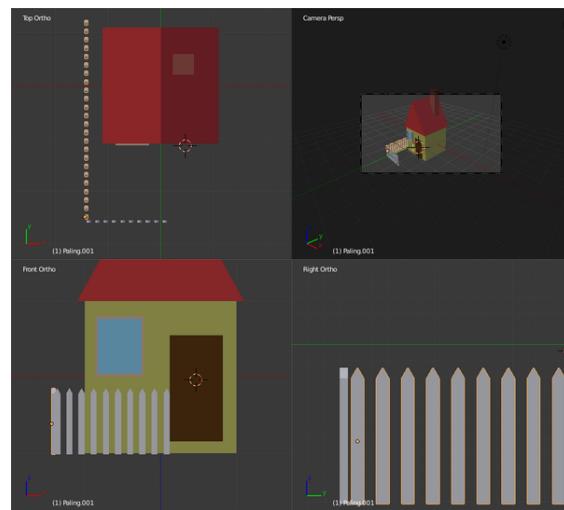
In Top view move the paling to the intersection of two grid lines at a suitable distance from the house.



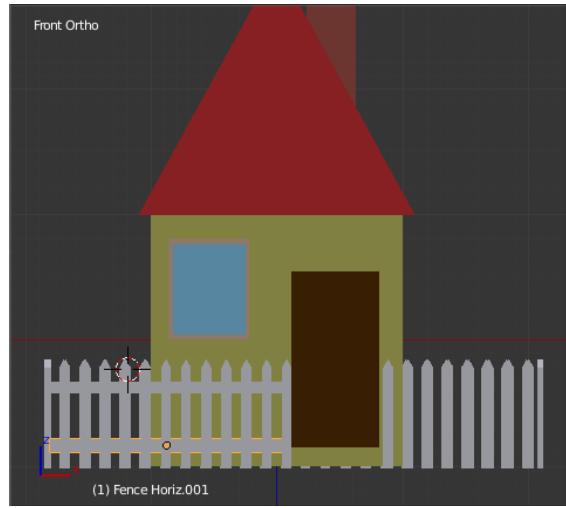
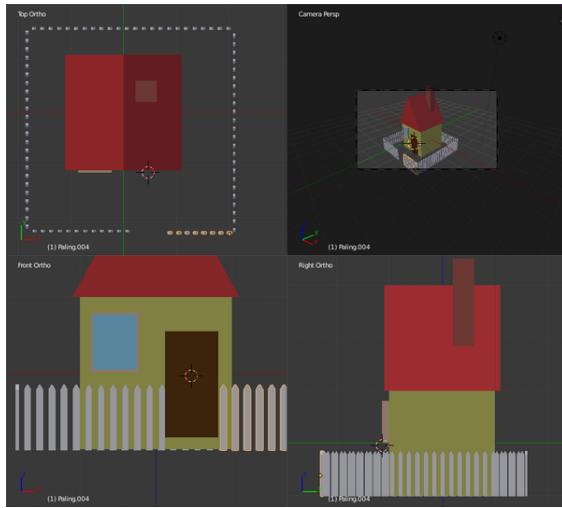
array count as necessary to get the right length.

Once more, type ALT + D to duplicate this third row, followed by X to constrain its movement parallel to the back of the house. Move the duplicate to the end of the row, and rotate it to be parallel to the side of the house with R Z –90 ENTER . Give it the same count you did to the array on the other side, so its length will be the same.

One last time: ALT + D to duplicate, Y to ensure it only moves parallel to the side of the house, move to the end of the row, rotate into position with R Z –90 ENTER .



You'll probably find it's so long it runs into the original row of palings you started with on the other side of the front of the house; reduce its array count to something like 8 and you should end up with a nice gap in front of the door.



Noob Note: I encountered a problem when rendering the fence. It became grainy, and when I say grainy, there are random clusters of black all over every rendered post. Some had few of this 'grain' while others were almost black. I have environment lighting at 1 and it's still goofed up. Would this be due to hand-sizing them instead of putting in all of the above values? I was unable to upload a picture of this to the site, so I used my Dropbox to share the link, [here](#).

Noob Answer: I don't think hand sizing would make any difference. The first thing I would do is go back into edit mode on the original paling and select all vertices. Then I would hit 'W' and click remove doubles. After that, recalculate the normals by holding 'Ctrl' and hitting 'N'. Those operations typically fix common rendering issues for me, hopefully it will work for you too.

Alternate Noob Solution: I encountered this problem and it turns out it is caused by having more than one copy of the fence in the exact same place. Try moving the affected fence pieces, to find the duplicate then delete it. Remove doubles did not work for my instance of the problem.

Make the Horizontal Bars (Left Side)

The fence needs horizontal bars to hold the palings in place: be in Front view

LMB below the middle of the left-hand fence.

Create a plane, rotating it to the vertical as necessary (R X 90 ENTER) and scale it to the length of the fence.

Scale it again to 0.05 along the Z axis and move it into position. (if you keep CTRL pressed while you move it, it will snap to the grid.)

Go into Edit mode (TAB)

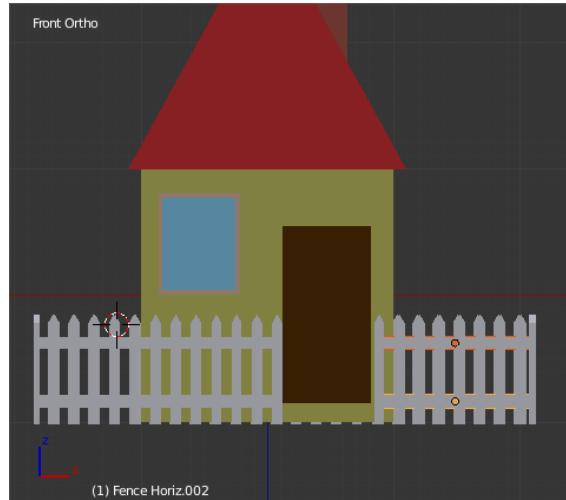
In Side view move it so it is against the back of the fence

and extrude it by -0.04 along the Y axis.

Go into Object mode (TAB). Give the horizontal bar the same material you previously created for the palings.

Duplicate it (ALT + D) and move it (keeping CTRL pressed) along the Z axis (Z) to make the bottom bar.

Make the Horizontal Bars (Right Side)



To make the bars on the right-hand part of the fence: In Front view, select *both* the horizontal bars you just created: RMB on one, then SHIFT + RMB on the other (you may need to zoom in to be able to select them without accidentally including the rest of the fence). Duplicate them with ALT + D . Move your duplicates along the X-axis X until they are horizontally centred around middle of the right-hand fence; now scale them horizontally S X until they are the right size.

You could have done the bars one by one, but it's quicker to do them both at once, don't you think?

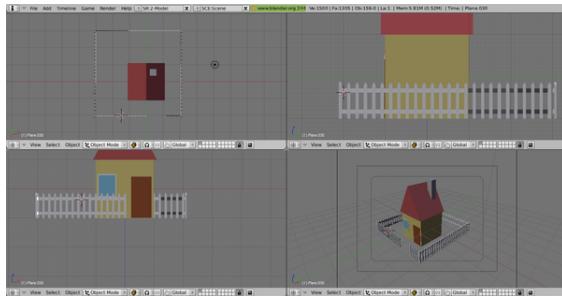
Put the Horizontal Bars Around The House

Select the two horizontal bars you created above for the front fence (make sure you don't select anything else, like the palings). Press ALT + D to make a duplicate, then R Z to rotate the duplicate about the Z-axis, type 90 for the angle and press ENTER to confirm the rotation. In the top view, move the duplicate (G) you just made so it's against the side fence. Keep scaling it along the Y-axis (S Y) and/or moving it along the Y-axis (G Y) until it's the right length and position.

Having done the side of the house, press ALT + D to make another duplicate of the horizontal bars, and R Z 90 ENTER to rotate it into the right alignment for the back fence. In the top view, move (G) it against the palings. Now keep scaling and moving it along the X-axis until it has the right length and position.

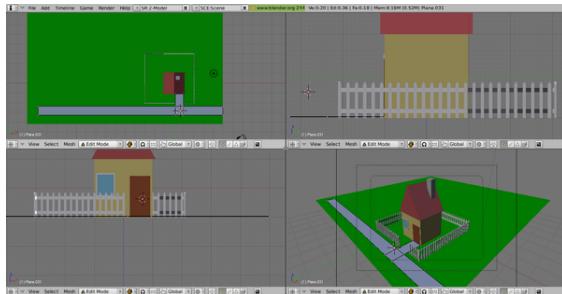
Having done the back of the house, do the other side: another duplicate of the bars, rotate 90° around the Z-axis, move against the palings, move and scale until they're the proper length in the right position.

Again, another duplicate, rotate and move/scale into position against the palings on the other side of the front of the house.



Your house should now look like this.

2.44.9 The Ground Plane and a Path



Make the Ground

In Top view put the 3D cursor somewhere near the middle of the plot, (click LMB)

Create a plane

Be in camera view (bottom right-hand window)

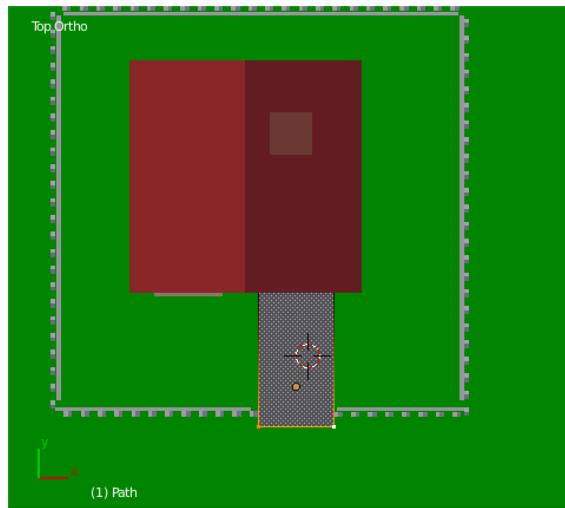
Scale it to extend just past the boundaries of what the camera sees (10.0 seems a good number).

Go back into Object mode and change its (object, mesh) name to "Ground",

create a new material for it, rename it "Grass" and select a suitable green from the colour selector ("00cd00" green3).

Check in Front view that it is level with the bottom of the house and fence and move it (along the Z axis) if necessary.

Make the Path



For the path: Go into Top view, create a plane.

Scale it to fit the gap in the front fence with some space on either side and position it just outside the fence.

Check, in Front view, that it lines up with the door and is level with the bottom of the house.

Scale it along the X axis to the same width as the door.

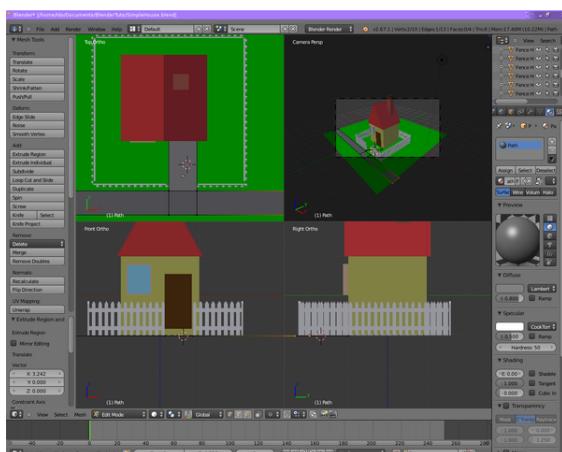
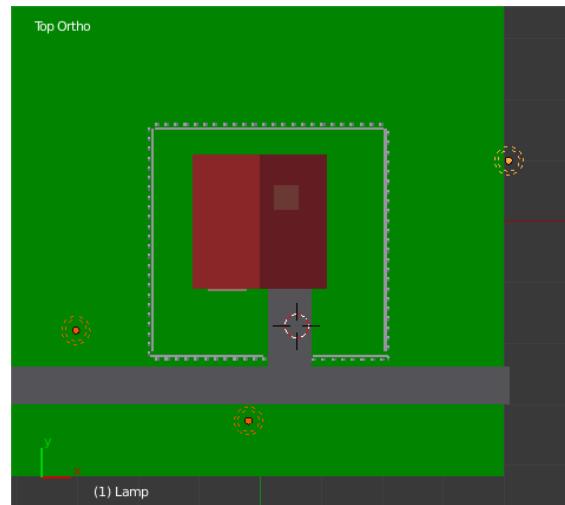
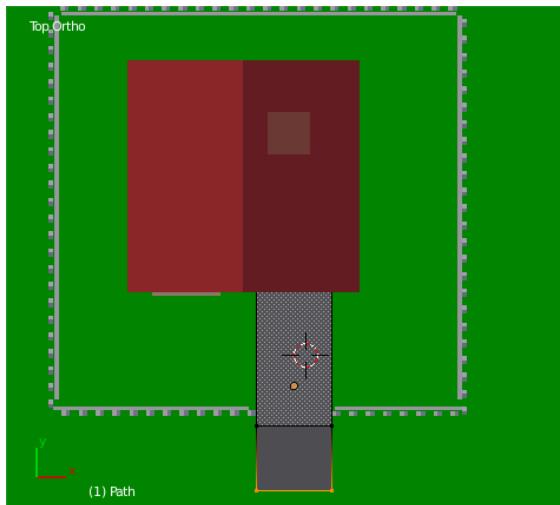
Be in Top view, Select the two vertices closest to the fence, and extrude E away from the house Y to make a suitable width for the footpath outside the fence.

Of those four vertices just outside the fence, select the left two and extrude a suitable distance to the left, and then the right two and extrude a suitable distance to the right, to make the path passing the house.

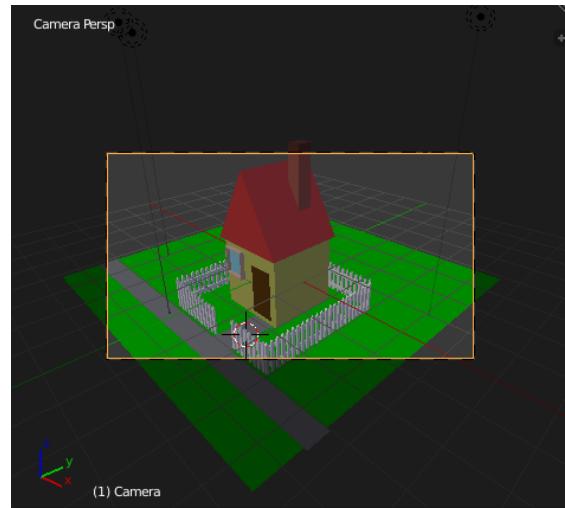
Give the Path Some Thickness

The path is still just a flat plane. Let's give it some thickness, so it rises slightly above the ground.

Go to Object mode. Make sure the path is still selected.



Go to the Modifiers Context , click Add Modifiers, and look for Solidify. The default thickness (the top-left editable field) is 0.01; change this to -0.01 so the path protrudes above the ground rather than into it.



2.44.10 Improve the Lighting

If you were to try to render now, you will probably find that most of your beautiful house is lost in gloomy shadow. To fix this, let's add a bit more light to the scene.

In the top view, find the existing default Lamp object. Duplicate it a couple of times, and position your copies around the house to give it a more even lighting.

area of the scene that will be rendered. If you RMB on the rectangle, you can reposition the camera from side to side or up and down with G, or do G Z Z to move it closer to or further away from the scene.

2.44.11 Check the Camera View

Before doing the render, check your camera is properly positioned to capture the beauty of your work. In the camera view (top-right quadrant of the split view), you should see something like this; note the *passepaport* (darkened area) outside the rectangle denoting the visible

2.44.12 Final Render

Now hit F12, and hopefully you should see something like this!



2.45 Pipe joints

How to model pipe joints.

2.45.1 T joint

Start in top view, and delete the cube.

- Add a cylinder object (*Add->Mesh->Cylinder*)
- 16 Vertices, No Caps, Depth 4 (For recent versions of Blender create the cylinder then press F6 to get these options)
- Switch to front view (*View->Front or NUM1*).

The X axis now points to the right. We will work along the X axis from now on.

- Change to Edit mode (TAB).
- Create a Loop Cut in the middle of the cylinder (**Ctrl-R**).

Make sure that the 3D cursor is placed in the middle of the cylinder.

- Switch to right view (**NUM3**)
- Box select (**BKEY**) the 7 vertices at the front
- Rip the mesh with **V**.

- Box select the vertices of the gap and extrude “edges” along the X axis.

- *Pivot -> Median Point (CTRL+,KEY)*
- To flatten the front: scale at the X axis by factor 0. (**S-X, 0**).
- Use the *Subsurf* modifier in combination with *Set Smooth*, *LoopCuts*, and *Crease* to get a nice transition.

2.45.2 6 cylinder joint

Be in Object mode (have nothing else selected)

Add a cube.

Switch to Edit mode

Select all, then Subdivide the cube once (**W, 1**) (**1**).

Select the 12 vertices in the middle of the cube’s edges.

Scale by factor 1.4142 (press **S** key and type in the factor with the keypad) (**2**).

Select all vertices, and Extrude “Individual Faces” (**Alt+E -> individual faces**) (**3**).

Select all vertices, and *Remove Doubles*.

Select the center vertex of each tube, and delete them to open the ends

Select the original vertices in the middle of the object and delete them (**5**).

Clean your mesh as described above.

2.45.3 3 cylinder joint

This is very similar to above, only the selection for scaling is different.

- Select the middle vertices at the edges where the cylinders shall come from. Additionally select the middle vertices at the faces that shall be kept free. You have to select 12+3=15 vertices.
- Scale your selection by 1.4142.
- Select the faces that shall be extruded.
- *Extrude->Individual Faces (5)*
- Select all, then *Remove Doubles*.
- Delete the inner vertices (6, 7).
- Select 3 vertices (which were middle of faces that were kept free) and 1 corner vertex in middle of the free faces (8).
- Make sure your 3D cursor is at the origin (and that you added the cube at the origin)

and switch to 3D Cursor Pivot (**.KEY** in v2.49b), then scale (**SKEY**), and type in 0.7071 (inverse of square root of 2, or Sin(45 degrees) or Cos(45 degrees)) and **ENTER** (**9**).

- Make sure you switch back to Median Point Pivot (**SHIFT+,KEY**) to straighten out pipe ends (**10**).

Clean your mesh as described above.

If your joint looks odd, it might be because of some edges in the mesh. Delete those by selecting it and delete (**XKEY**) edges (not vertices).

2.45.4 T joint with smaller diameter

We're going to create a structure like in **Fig. 4a** and extrude the inner circle. To do that we place a circle with *Retopo* ("Snap during transform") on the cylinder and join both objects afterwards.

- *Add->Mesh->Cylinder*
 - 8 vertices (of course you can use more vertices if you like)
 - *No Caps*
 - *Depth 4*
 - **Noob note:** More vertices in your cylinder will mean more intuitive work and figuring [more loop cuts, face making, etc.] later on when retopo-ing. I suggest following the tutorial first,

then try to do a more advanced model with more faces / vertices when you understand the tool. If you have trouble [as I did] getting really nice geometry, do a search for some tutorial videos. There are many really helpful ones.

- *Add->Mesh->Circle* (**Noob note:** To make sure you create these as separate objects, be in Object Mode (not Edit Mode).)
 - 8 vertices
 - *No fill*
- Rotate the circle by 90° at the Y axis (**R->Y->90**).
- Move the circle in front of the cylinder(**G->X->2**).
- Switch to side view (**Num-3**).
- Scale the circle to the diameter of the smaller cylinder.

(**Noob note:** I felt this was a little unclear. What the author means is, scale the circle down until it is the size of the diameter of the smaller cylinder you are going to create from it later by extrusion. You would be wise to scale it so it fits (straight backwards from view) inside the two closest faces of the cylinder as per **Fig. 4a** if you wish to follow the tutorial.)

- Switch to edit mode, select all vertices of the circle.

(**Noob note:** To recap: you should be in edit mode for the circle, with the all of it selected, with the cylinder object directly behind it in from your viewpoint in orthographic mode.) Make sure that you don't work in Wireframe view.

- Click on *Snap during transform*, that is the icon of the magnet in the 3D view header, select "Face" and after that a third box will appear, select there "closest". After that, push: "num3" and then "G key" and reposition it a little. After that you will see it lies perfectly around the cylinder.

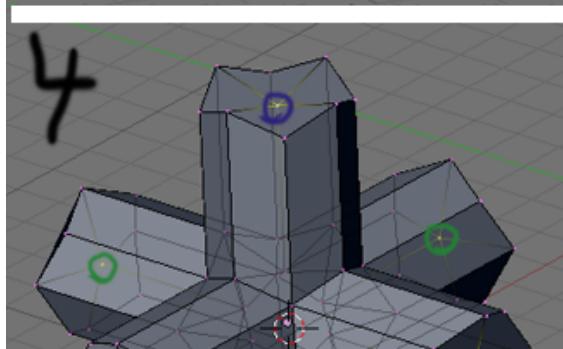
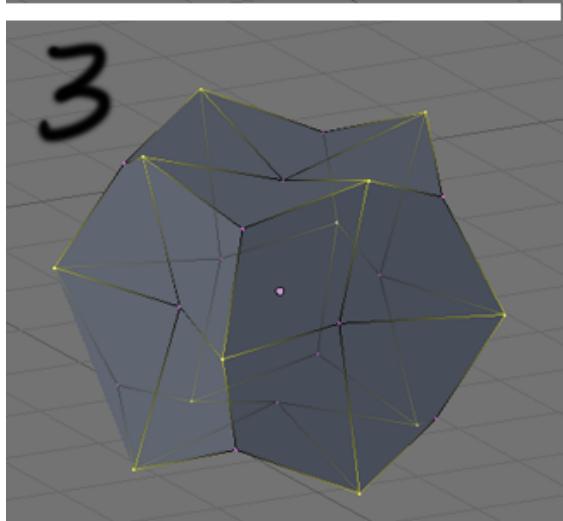
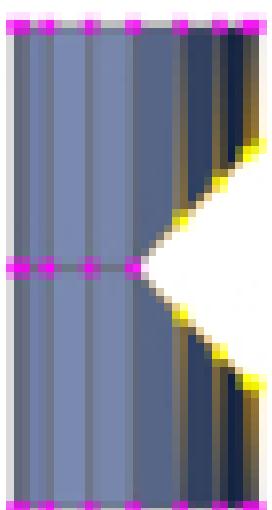
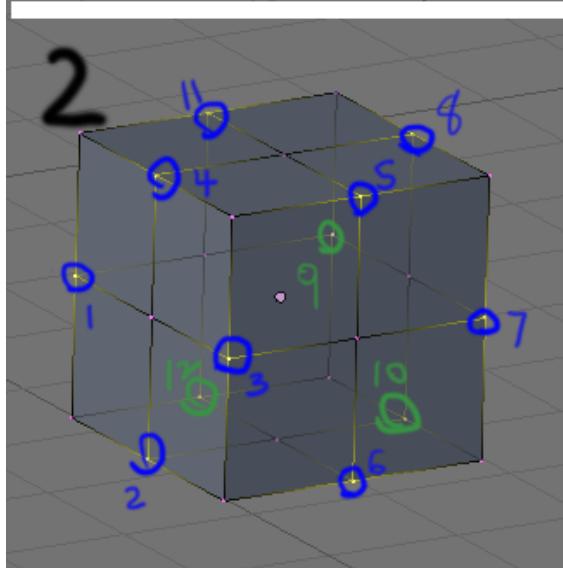
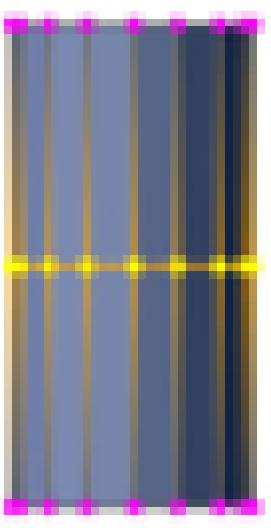
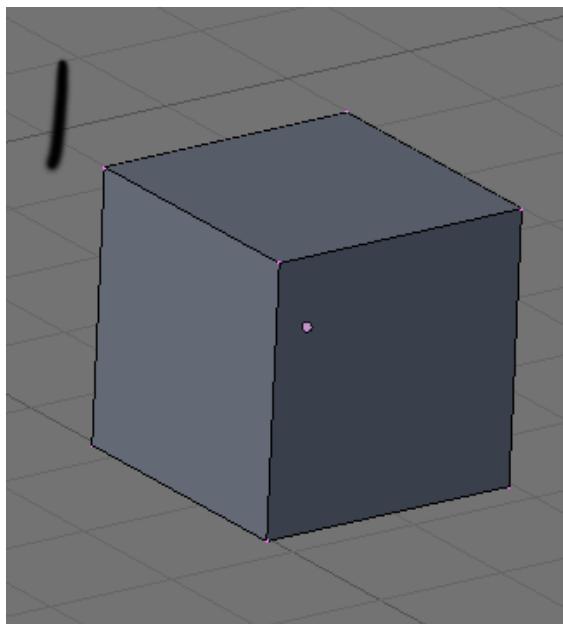
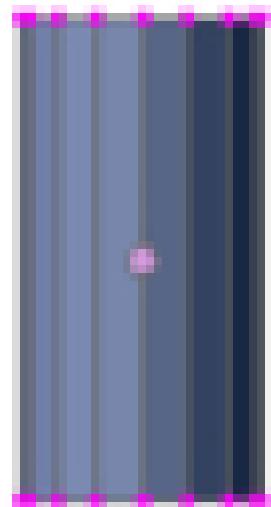
(**Noob note:** What "retopo" (Snap during transform) does [*in this case*]; is project the selected topography of an object in edit mode directly backwards from your selected view point onto the topography of another [seperate] object. So you must be lined up view-wise for this to work. I've also found sometimes I must press the **Enter key** for the "Snap during transform" to take effect even if I've selected it with the **LMB**.)

If the vertices of the circle are not adjacent to the cylinder but have jumped to wrong places undo the last step (**Ctrl-Z**), deselect and reselect all vertices and try again. This does help (strangely). *Snap during transform* works with limited accuracy, if the vertices don't fit perfectly you have to move them by hand.

- Change to object mode, select additionally the cylinder and join both objects (**Ctrl-J**).

Now you work best in *Wireframe* view.

- Add three loop cuts (**Ctrl-R->3**)
- Delete the middle vertex.
- Connect the free vertices (**F**).
- Select the circle.
- Extrude and clean up.



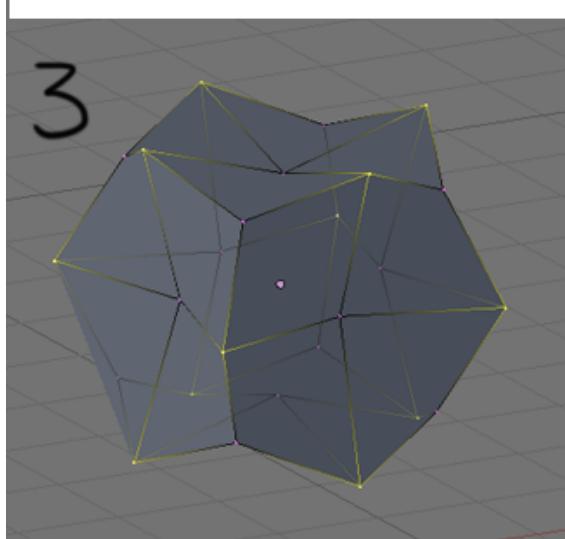
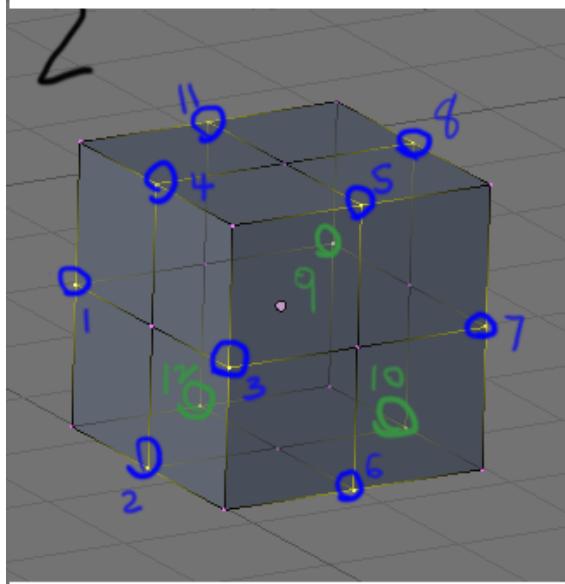
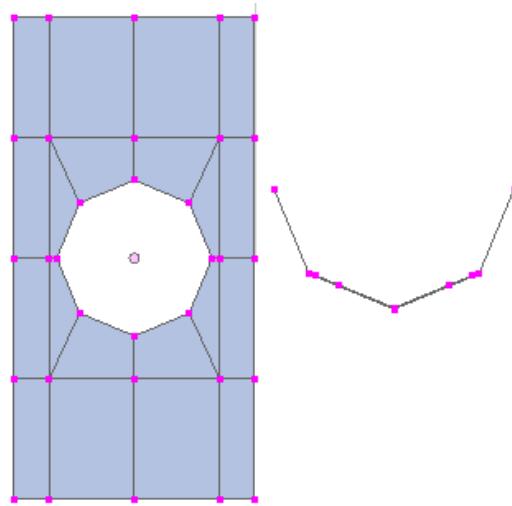
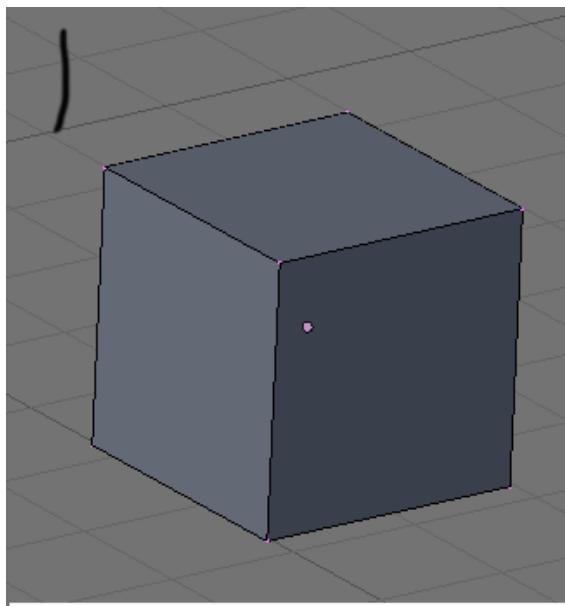


Figure 4a: Basic structure for a T joint with a smaller diameter

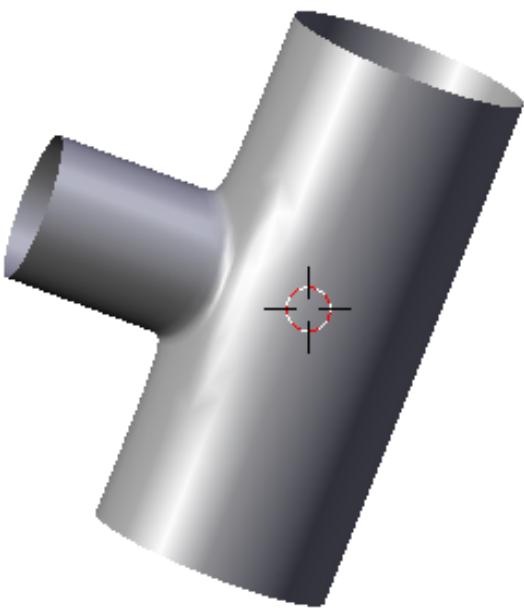
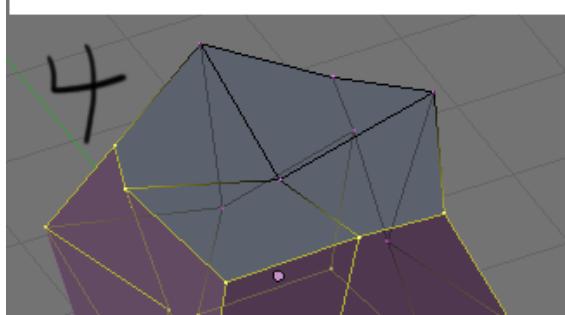


Figure 4b: The result



2.46 Lighting Suzanne: Introductory one lamp lighting

2.47 Lighting Suzanne: Introductory one lamp lighting

If you have followed all these tutorials you now have quite a library of images: A simple figure with a hat, a penguin, a volcano, a jeep and several goblets, not to mention dice, a head and a house! To show them off to best effect, you need to experiment with lighting.

Lighting is a complex subject, as any photographer can tell you. Blender allows many different arrangements and numbers of lamps to be used in combination to light your image. For this introductory tutorial, let us consider single lamp lighting and do some experiments to see how it works. (In later sections more complicated lighting is covered: [Lighting Rigs](#).)

2.47.1 Suzanne, Our Star

We need an object to light. Who better than Blender's favorite resident simian, Suzanne! Open the default startup page. Delete X the default cube. Be sure the 3D cursor is at the origin. Add a plane. *Add>Mesh>Plane*, scale it up nice and large S 2KEY 0KEY , and in the Properties panel (which by default is the lower right panel) add

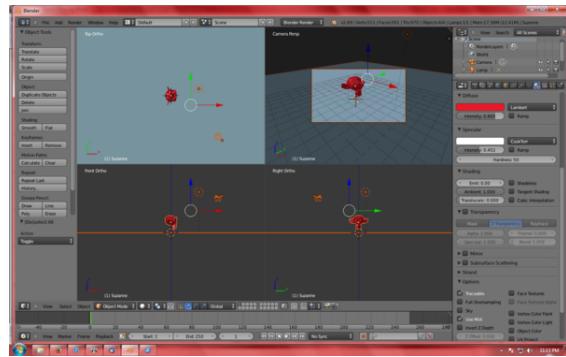
a material  color such as light blue that will render well. In that same panel be sure that under SHADOW the Receive box is checked.

Now Add Suzanne, the famous monkey. *Add>Mesh>Monkey*. Move her up about 1.5 Blender units so she is above the plane but close to it. G Z 1KEY .KEY 5KEY . Rotate her around the Z axis so she is facing the Right View. R Z 9KEY 0KEY . Add a material  color such as red (red is my favorite color).

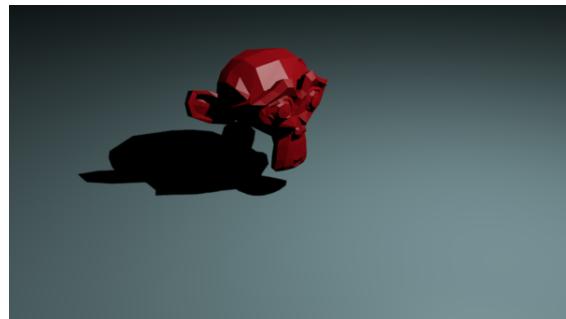
2.47.2 The default light

Now lets change to the Quad view, which you should refer to extensively throughout this tutorial. Press CTRL ALT Q to bring up this view. You now have the TOP, FRONT and RIGHT ORTHO views. In the top right you will see your camera view. Suzanne should be seen in 3/4 profile just above the plane. If not, rotate and scroll in or out until she is. Then I suggest you bring up the object panel N , scroll down to View and put a check next to Lock Camera To View.

In the outline panel, select Lamp. You will see the default lamp position. In Ortho Quad view, we can see that the default lamp is up, in front of and slightly to the right of Suzanne as she now faces. If you don't see it, scroll out



Default Quad view



First Render

in each view window until the lamp shows up as a point with a double dashed circle around it.

When your screen looks like the screenshot, render for the first time to get a good image of Suzanne and her shadow.

2.47.3 Types of lamps

Lamp should still be selected in the Outline panel. In the Properties panel, click the  button to see the properties of the light. The default lamp is a POINT lamp with ENERGY 1.000. Let's experiment with those settings. First, click in turn each of the different kinds of lamp, SUN, SPOT, HEMI and AREA, and render after each one. You can see that each of the light types has its own characteristics. With the light in the default position, your images should look like this:



Comparison of light types.

Point lamp

This is the default type. The light rays are assumed to originate in a single point and spread from there. At

the default value of ENERGY 1.000 is not very bright and covers a limited area. It casts a good sharp shadow. Point is the basic general purpose lighting in Blender. Advanced lighting rigs will frequently use several point lights with different settings.

Sun

The sun is assumed to be infinitely high in the sky so its rays are all parallel. It puts out a bright light and casts heavy shadows. It is the lighting type of choice for outdoor scenes but can also be used to good effect indoor.

Spot

The Spot lamp behaves very much like a theatrical spot-light (DUH). It casts a limited circle of light. Within the circle it is bright and casts a heavy shadow, but outside the circle all is dark. (Note: Under the Shadow setting, the Spot lamp is the only one that can cast a Buffer shadow. The distinction is subtle but important in some more advanced applications.)

Hemi

The Hemi lamp resembles lights photographers choose for indoor shots. It is the only kind of light that does not cast shadows. It gives a very bright lighting that is relatively even over the entire picture.

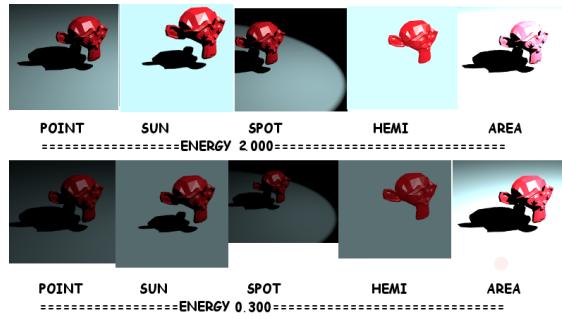
Area

This light behaves more like a flood light, as if the area is flooded with it. It is extremely bright and casts a very heavy shadow.

2.47.4 Light Energy

Each type of light has its own brightness. This is controlled mostly by the ENERGY setting of the light. Obviously less energy reduces the amount of light, more energy increases it. Still in the Properties panel, try different settings for ENERGY and see how it affects the image.

Begin with energy increased to 2.000. Render each type of lamp. We can see that reacts differently to the energy settings. Area is almost too bright at Energy 2.000, while Point and Spot are just reaching nice bright levels. Sun is ... Sun. Remember that it is infinitely far away and its rays are parallel. Little changes with the Sun. Now reduce energy to 0.300 and render each type of lamp. Point and spot at these settings are way too dim, while Area becomes a very useable setting.

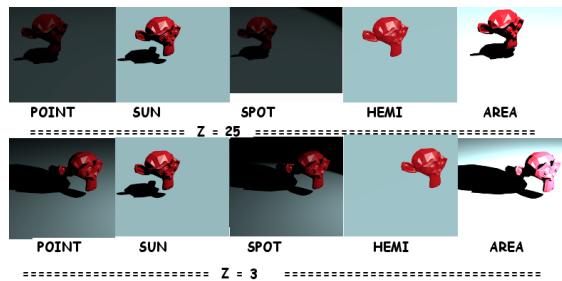


Lighting varying energy settings

2.47.5 Distance and Height

The distance of the lamp also affects brightness and its position affects several things, particularly including the shadows. In the quad view, grab the default lamp and raise it up quite high, say to Z=25. The light falls off with distance, so point becomes very dark. The spot area increases with distance so the circle in the spot example is now quite large. The Sun ... well, the sun was infinitely far away already, remember? The shadow cast by the sun does not change, and hemi still casts no shadow. But the shadow cast by the other three lights moves considerably and is now almost directly under Suzanne.

Lower the lamp to Z=3, placing it almost directly at a level with the subject. Both the distance and the angle of light change. The brightness increases, the size of the spot circle shrinks, and now the shadow is long and far behind the subject, except, of course, for the Sun which remains infinitely far away and casts parallel rays.

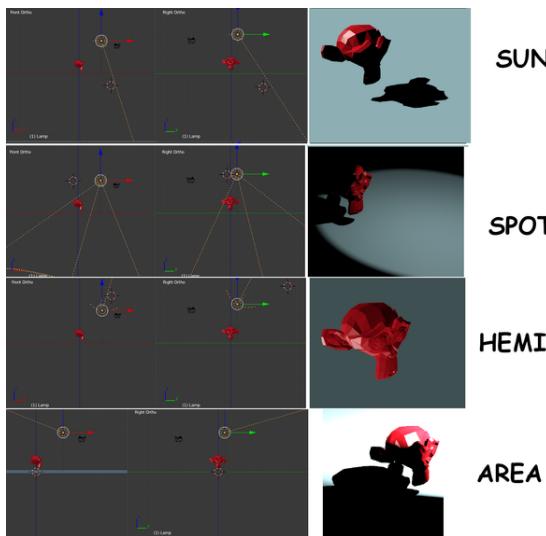


Effects of height on lighting

2.47.6 Rotation

The last setting we will consider in this lesson is Rotation. The Point light cannot be rotated since it is equal in all directions. Each of the others reacts differently to being rotated.

The Sun finally shows a different effect when rotated. I am not sure how Blender interprets the difference between location and rotation for the Sun. It seems rather counter intuitive. The sun when rotated seems to me to behave as if it has been moved. But in any case you can see that the



Effects of Rotation on lighting

shadow and the lighting show a distinct difference.

The Spot behaves very much like a real spotlight. Its circle of light moves with the rotation and frames whatever it is pointed at.

The Hemi lamp also finally shows a bit of difference. Even pointing it entirely way from the subject still lets light leak through it and give that same shadowless flat lighting.

The Area lamp has its characteristic brightness over a large field, and then suddenly cuts off in to total blackness. The contrast is very striking.

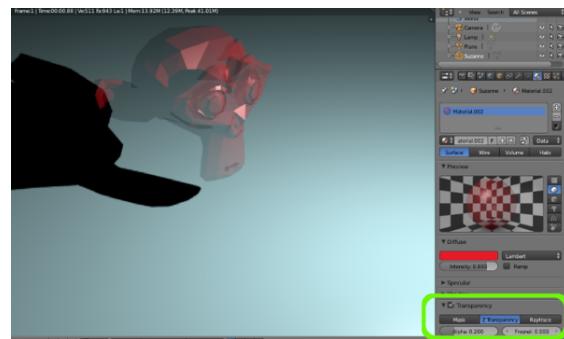
2.47.7 Experiment!

You can experiment with other placements of the light, and see the effects on the illumination and the shadow. There is an almost infinite variety of ways to light a Blender image, and this is just with a single lamp. Multiple lamp setups increase the variety many fold as will be seen in the Advanced portion of this tutorial.

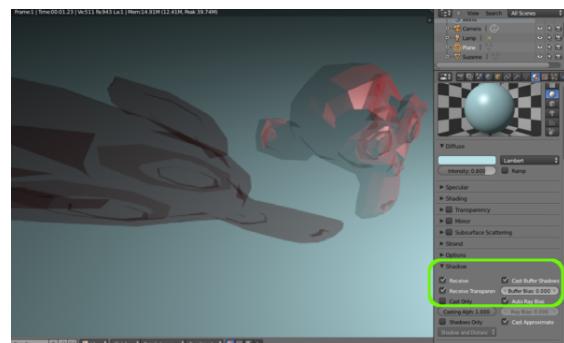
2.47.8 Note on Transparency

A special case of lighting is transparent objects. What sort of a shadow do transparent objects cast? You can try an experiment. Click on Suzanne to select her. Now

in the Material panel scroll down to Transparency and put a check in the box. Just below that is the Alpha setting, which controls the amount of transparency, with 1.000 being fully opaque and 0.000 being completely transparent to the point of invisibility. Lets try setting it to 0.200 for an almost glassy effect. Render your view, and then ask yourself, why does this glassy object still cast a thick black shadow in the render?



Where does that shadow come from?



Plane Shadows set to Receive Transparency

The answer is not in Suzanne, but in the settings for the plane which is her background. Recall that we checked Receive under shadows for that plane. But go back to the properties panel for the plane, and notice that there is another check box under Shadows that says Receive Transparent. By default, to save rendering time, Blender assumes that all objects block light equally. Only if that box is checked does it make a check for transparency in the object casting the shadow. Now render your view and you will see that the shadow is suitably glassy.

Experiment with this one. Try different types of lamp (Area is so strong that transparent objects effectively disappear) and different values for the alpha setting. You might want to remove the color material from Suzanne and see what effect that has.

2.47.9 Note on World Lighting

It is not absolutely necessary to use any lamp source. In the World panel you will find several lighting settings that simply produce light throughout the scene without any particular source. Experiment with Environment Lighting and Indirect Lighting and see the effects they produce.

2.48 Overview

2.48.1 Why Use Curves?

You've learned how versatile mesh objects can be, and how they can easily produce flat surfaces with sharp edges and even give a convincing representation of curved surfaces with rounder edges. So why do we need a separate kind of curve object at all?

There are important reasons.

- The various kinds of curves and surfaces/patches (Bézier, NURBS etc) were used in computer graphics before meshes were developed. In the days when memory was more expensive, they offered a more compact way to represent complex shapes. Even now, they are still common in CAD and other technically-oriented graphics applications, where they make it easy to precisely specify the shape of a curve. Thus, if you are importing data from such applications, you will need the ability to represent such curves—even if you end up converting them to meshes before applying materials and textures for rendering.
- You can use curves as guides for shaping meshes. This makes it easier to construct certain kinds of complicated curved shapes. Particularly since it is simpler to make changes to the curve (with fewer control points), than after it has been converted to a mesh. To this end, Blender defines *scaling radius* and *tilt angle* settings for curve control points, which make no difference to the appearance of the curve itself, but have an effect when it is used to deform another shape.

2.48.2 Bézier Versus NURBS

In the following pages, you will come across two kinds of curves/patches.

Bézier objects only occur in Blender as curves, while NURBS (“Non-Uniform Rational B-Spline”) objects can be curves or surfaces. If you've used 2D drawing programs like Inkscape or Adobe Illustrator, you would have come across Bézier curves before. NURBS curves are a mathematical generalization of these, which are heavily used in CAD applications. Unlike Bézier curves, NURBS curves allow the specification of a variable *weight* for each control point, which governs how closely the curve passes to that point.

2.48.3 More Than You Wanted To Know About Curves

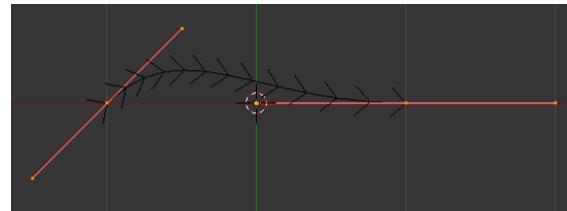
As usual, Wikipedia has the lowdown on all the interesting trivia about

- Bézier curves
- NURBS curves

2.49 Intro to Bézier Curves

Note: Some pictures are outdated.

2.49.1 Bézier Curves



It should look like this.

- First start a new Blender project, and delete the default cube.
- Press: SHIFT + A → Curve → Bezier to create a new curve. Switch to top view NUM7 for a clearer look. You may want to zoom in a bit as well. TAB into Edit mode.

The black line with the extra angled lines like centipede legs coming off it is the Bézier curve. The white or orange dots are the *control points*, with the ones in the middle of the pink handle lines defining the endpoints of the curve segment.

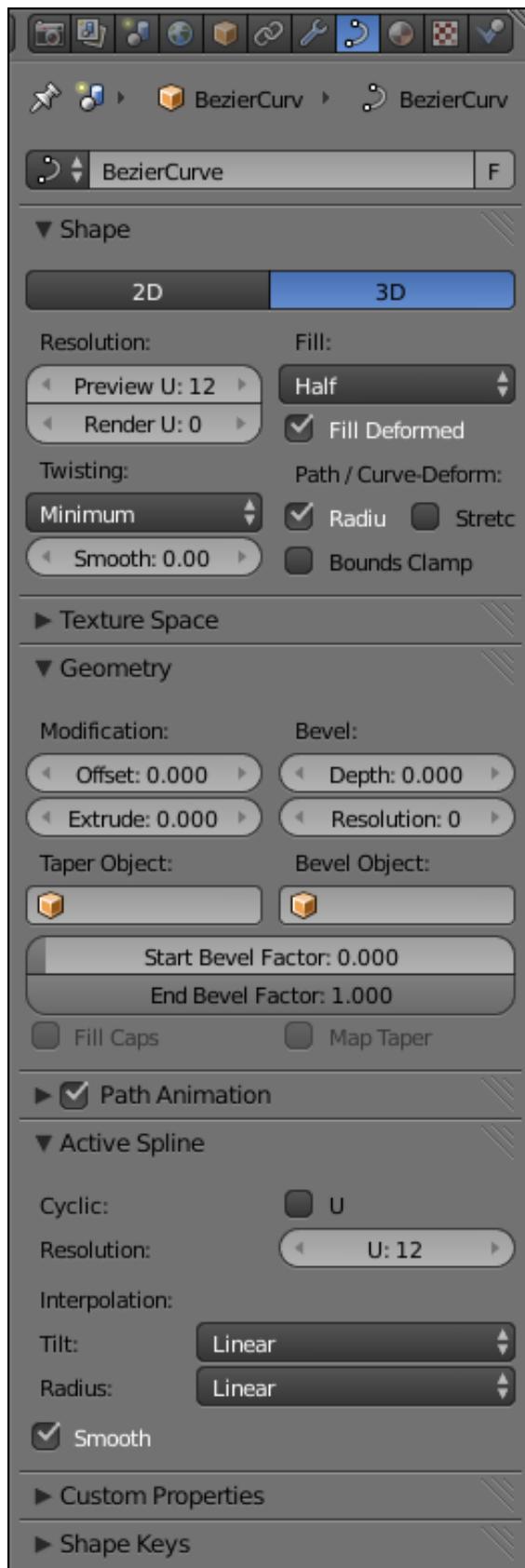
Press A to deselect the selection, so that all the control points turn black, and the lines connecting them turn red. You can RMB on any point to select it; however, selecting a curve endpoint selects the entire handle line passing through it.

You can move a selected point in the usual way, with G : note how moving an endpoint causes the curve to bend so it always connects to the endpoint. Moving just a point at the end of a control handle affects the inner part of the curve, making it bend more or less sharply away from the endpoint: try moving one of these points around, and note how the handle gets longer or shorter, and automatically rotates as necessary to remain a single straight line.

Alternatively, you can select an endpoint (which selects the entire control handle, remember) and use R to rotate the handle, and S to make it longer or shorter.

Resolution

In the Curve Context  in the Properties window, further down you will see the Active Spline panel. (This is



only visible in Edit mode.) Note the editable field next to the label “Resolution:”, probably showing “U: 12”. That number governs the number of straight-line segments that

the curve is converted into for rendering purposes; the more of these, the smoother the result, but it will add a bit to the render time.

This resolution setting is also used when converting the curve to a mesh (ALT + C).

Extending Your Curve

So far your Bézier curve only has one segment. You can add segments in several different ways:

- Back in EDIT mode, RMB on an endpoint at either end of the whole curve, and use E to extend a new curve segment connected to the same endpoint. (We won’t say “extrude”, to avoid confusion with a different extrusion function discussed further down.)
- RMB on an endpoint at either end of the whole curve, and CTRL + LMB where the endpoint of the new segment will go; the new segment will immediately be added between the previous endpoint and the new one.
- Select the endpoints of a single curve segment, press W and choose “Subdivide” from the menu. This splits the curve segment into two new connected segments.

Note that if you CTRL + LMB when the selection is not precisely one endpoint of the whole curve, it will add a new handle that is not connected to any part of the existing curve. You can extend from this handle to create a whole set of new curve segments that are still part of the same curve object, even though there are no lines running between this piece of curve and the previous one. You can join two separate pieces of the curve with a new segment by selecting an endpoint of each and pressing F .

You can also add new pieces with SHIFT + A in Edit mode.

Removing Points and Segments

You can delete any control points by selecting them and using the familiar DEL or X . The menu that pops up asks you if you want to remove the “Selected” points, the “Segment”, or “All”.

- “Selected” removes all points that are part of the same handle(s) as the selected point(s). Removing an endpoint of a curve piece removes the connected segment. Removing an interior endpoint causes the replacement of the curve segments on either side of it with a new one running directly between their remaining endpoints. Thus, the curve piece of which that segment was a part remains a single curve piece afterwards.

- “Segment” removes a segment between two connected endpoints. If neither endpoint was an endpoint of the curve piece, then the remainder of that piece of the curve becomes two disconnected pieces.
- “All” removes *all* of the control points, leaving an empty curve object!

Handle Types

So far, all the handles on your curve have been *aligned*. This means that adjacent curve segments are guaranteed to join smoothly. If you select a control point and press V, you will get a menu allowing you to set other types for the handle:

- Free* means the two parts of the handle on either side of the endpoint are free to rotate independently, so the two curve segments can now meet at a corner.
- Vector* means the selected part of the handle points at the other endpoint of the same curve segment, and will keep pointing that way even if you move either endpoint. If you set the part of the handle from the other endpoint pointing back this way to “Vector” as well, then the curve segment becomes a straight line. However, as soon as you explicitly move the handle itself, it reverts to being “free”.
- Automatic* is like “aligned”, except Blender will automatically adjust the orientation of the handles as you move the endpoints, to try to keep the curve smooth. Moving either end of the handle will change it back to “aligned” type.

2D Versus 3D

At the top of the Shape panel in the Curve Context  in the Properties window, you will see buttons labelled “2D” and “3D”. “2D” means the points of the curve are constrained to lie in a single plane, while “3D” means they are free to be located anywhere in 3D space relative to each other. Initially “3D” should be selected.

Try moving one point along the Z axis. Now click the “2D” button, and you should see all the points you moved snap back into the same plane as the rest. Also the “centipede legs” along the curve should have disappeared (their presence indicates a 3D curve). You can alter the orientation of this plane by TAB bing into Object mode and then rotating R the whole curve about any desired axis.

Closing, Filling and Extruding

Each curve piece can be individually *closed*, meaning that an extra curve segment is automatically added between the two endpoints of the piece. This is governed by the

“Cyclic: U” checkbox in the Active Spline panel in the Curve Context . Each closed piece of a 2D curve is automatically filled to form a flat surface.

The curve can also be *extruded* into the third dimension, effectively turning it into a ribbon-like shape. Look for the “Extrude:” editable field in the Geometry panel, and put a nonzero value into that to specify the extrude width.

If you extrude a 2D closed curve, the flat surface becomes a solid object, and the result is a *prism*—a solid shape with a uniform, but arbitrarily complicated, cross-section.

Scaling and Tilting

Turn your curve into a ribbon by specifying a nonzero Extrude value. Change your view so it is not exactly from the top or bottom, but at some angle. Notice how the ribbon is of constant width and always perpendicular to the plane of your curve.

Select an endpoint, and press ALT + S. As you move the mouse, you are changing the *radius* of the endpoint from its default 1.0 value, and this causes a corresponding scale factor to be applied to the width of the ribbon at that point. You can see (and also edit) this radius value in the Properties Shelf N, in the Transform panel at the top.

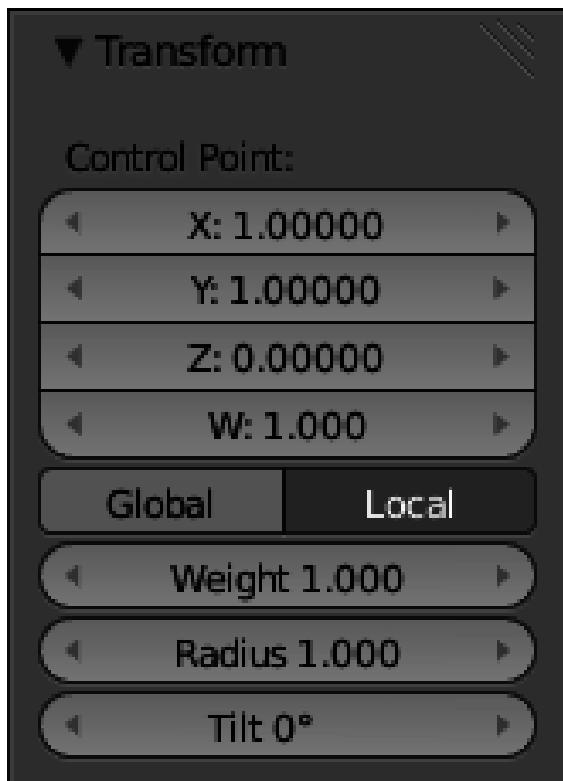
With an endpoint selected, now try CTRL + T (**Note:** This only has an effect with 3D curves, not 2D ones). Now as you move the mouse, you are applying a *tilt angle*, which correspondingly alters the angle of the ribbon at that point. This value is also accessible in the Transform panel in the Properties Shelf.

The scale radius and tilt become particularly useful when you apply a *bevel shape* to the curve—then you will start to get much more complicated shapes than simple ribbons.

2.49.2 NURBS Curves

Start a new Blender document, and get rid of the default cube.

This time, do SHIFT + A → Curve → Nurbs Curve, and TAB into Edit mode. The result looks a bit similar to a Bézier, except there are no control handles, only the segment endpoints. Instead of dragging handles, you can adjust a weighting assigned to each control point. Bring up the Properties Shelf with N if it is not already visible. If you have a single point selected, in the Transform panel at the top, you should see four editable fields under the heading “Control Point:”, labelled “X:”, “Y:”, “Z:”, and “W:”. The first three are of course the position of the point; the “W” value is like a gravitational field strength that attracts the curve to the point. Try adjusting this, and see how it influences the shape of the curve near that point.



You can move endpoints in the usual way, but there are no handles to rotate. All the other options for add and deleting endpoints apply as for Bézier curves: **CTRL + LMB** or **E xtend**, **W →Subdivide**, **DEL eting Select/Segment/All**. The curve can be 2D/3D, extruded and cyclic, and each endpoint has a **ALT + S**cale radius and a **CTRL + T**ilt angle. In the “Active Spline” panel in Object Properties, there is an additional “Endpoint: U” checkbox which forces the curve to pass through the endpoints (this is ignored if Cyclic is enabled).

SHIFT + A →Curve→Nurbs Circle inserts a NURBS curve with eight control points arranged in a square, “Cyclic: U” enabled, and the weights of the corner points adjusted so the curve is an exact circle.

SHIFT + A →Curve→Path inserts a NURBS curve with five control points in a straight line, and the “Endpoint: U” option already checked.

2.49.3 Which Curves To Use?

As mentioned earlier, NURBS curves are heavily used in CAD applications, while Béziers are popular in 2D drawing applications. If you are importing data from these applications, then you won’t have any choice about which one you end up with.

But if you are creating your own curves, then you have a choice. If you have done drawing with Bézier curves a lot, then you should feel at home with them. If you just want simple curves to deform a shape or guide an animation,

NURBS curves could very well do the trick—you could even leave the W values at the default, and just add, delete and position points to get a suitable curve.

2.49.4 Extruding from 2D to 3D

You may have noticed you can only modify the curve in two dimensions, and now it’s time to explore the third dimension! Extruding is where you define a two dimensional ‘profile’ shape, and it is ‘swept’ through space to create a volume.

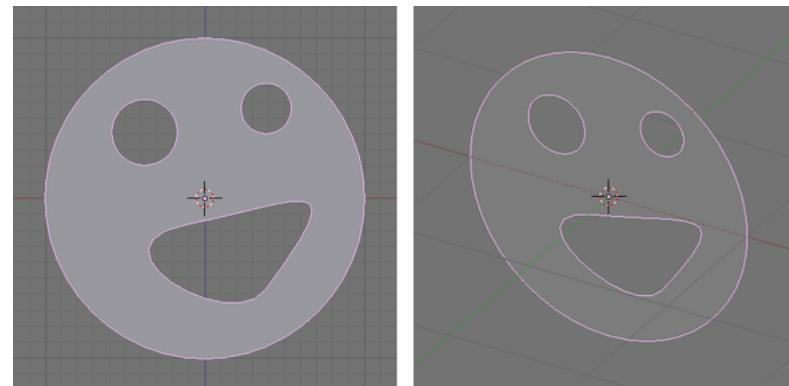
Noob note Bézier circles are not true circles, they are approximations. For artistic purposes, this may not really matter, but for precision modelling only NURBS circles should be used. This is due to the math used to describe the Bézier and NURBS curves.

2.49.5 Make a Simple Face of Bézier Curves

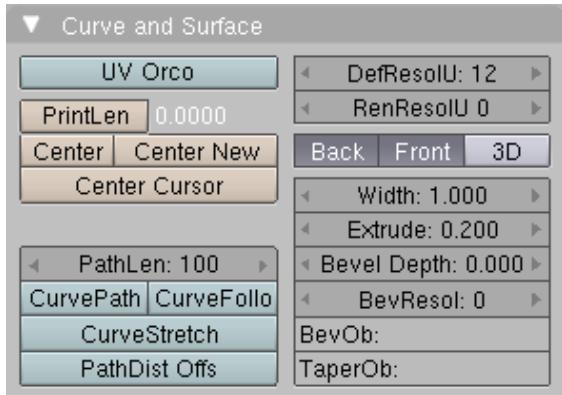
- **Make a curve and fill it:** Being sure you are in 2D mode, Use the steps above to create something simple, and fill it in using

ALT + C .

Noob note: you may find it easier to use Bézier circles, instead of filling discrete curves.

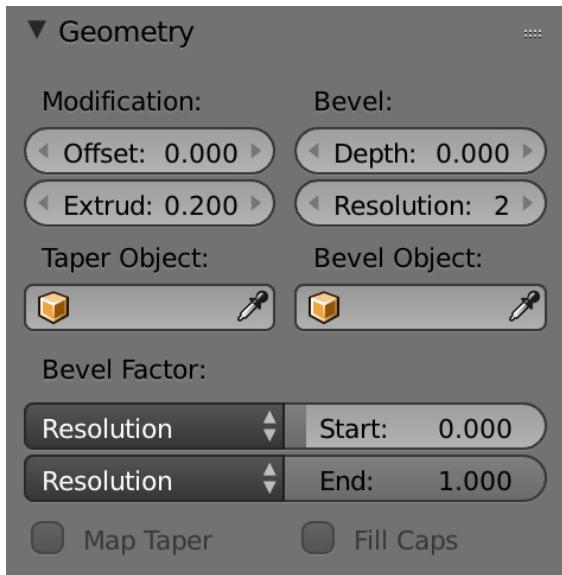


Note that the eyes and mouth of the face above are parts of the same curve object. Blender can only identify holes in a filled curve if the holes are part of the same object as the outer curve. If you had created four separate curves then filling the outer curve would have covered the eyes and mouth. To create a detached section of a curve, be in edit mode and deselect all the points (**A**). Then create new points (you will need two or three) with **CTRL + LMB**. Alternatively you can create the pieces as separate curve objects and join them later with **CTRL + J** in object mode. You might prefer the alternative way if you don’t want a lot of control handles and “centipede legs” to distract you.



Extrude the Simple Face

Set the extrude depth: Click on the Object Data button and find the Geometry panel. (Older versions: find the 'Curve and Surface' box in the Editing tab of the Buttons window.) There is a slider called Extrude. Set the Extrude depth to something other than 0, and probably less than 1.



Bezier Geometry

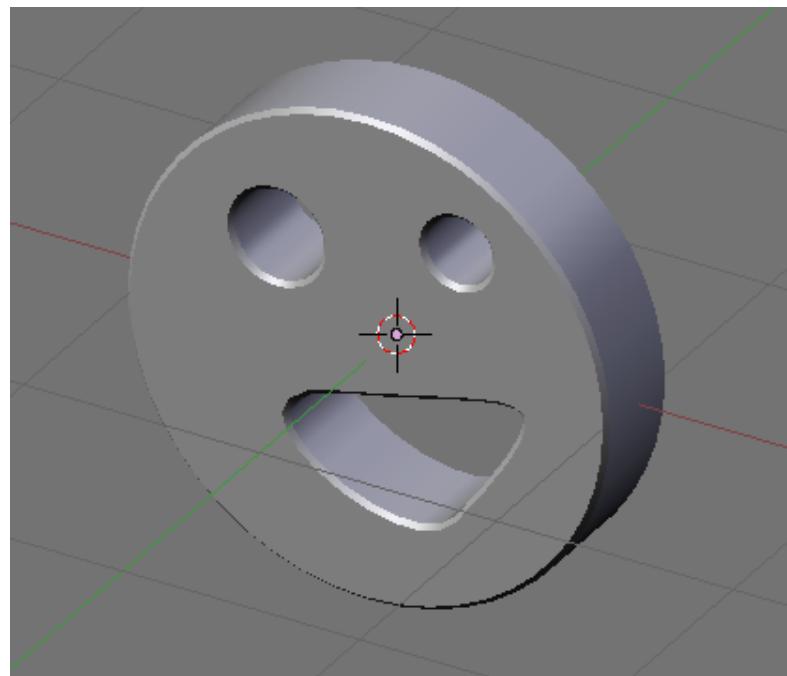
Behold, your 2d curve has transformed into a neat 3d structure. The great thing is, you can still edit the curve as if it were just 2d, and the changes will update in real time.

Bevel the Simple Face

Now you have an extruded shape, you should start playing around with some of the other curve settings on offer, so here is a description of how the Bevel depth and Bevel resolution sliders work.

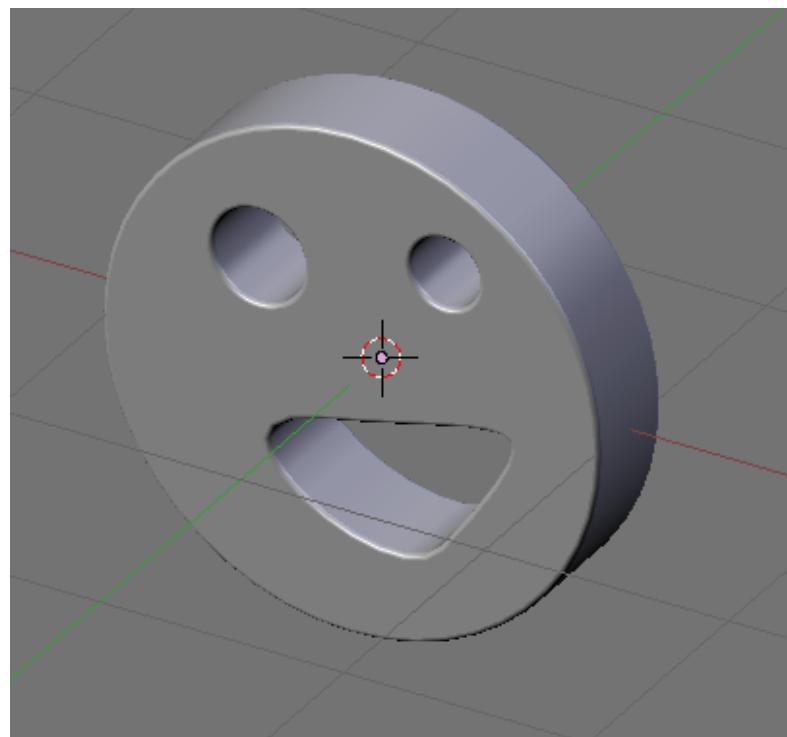
Try setting the Bevel depth to a small value, say 0.02. This will cut off all of the sharp edges, and give a bevelled

effect all around the shape.



As you may guess, Bevel resolution decides how many times the algorithm divides up an edge. Higher values than 0 result in smooth curves rather than sharp edges, but dramatically increase the number of vertices in the shape.

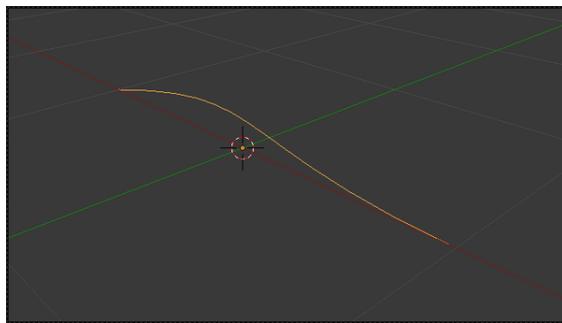
Try setting the resolution to 3 or 4, you should see an effect like this:



2.50 Bevelling a Curve

A simple, non-extruded curved line on its own will not be visible when rendered in Blender. Things are different, however, when you *bevel* the curve—that is, use some two-dimensional shape as a cross-section, and the line curve becomes a guide for extruding the shape into the third dimension.

2.50.1 Built-In Bevel



Bézier curve with default settings

Start with a curve object—any curve will do. Here we use a Bézier curve.

Look in the Curve Context in the Properties window, for the Geometry panel. There you should see two editable fields with the title “Bevel:” above them, one labelled “Depth:” and the other “Resolution:”. Try setting the depth to something like 0.1.

Now your curve is no longer a simple line. It should have a V-shaped cross section, perhaps like a piece of bent angle iron.

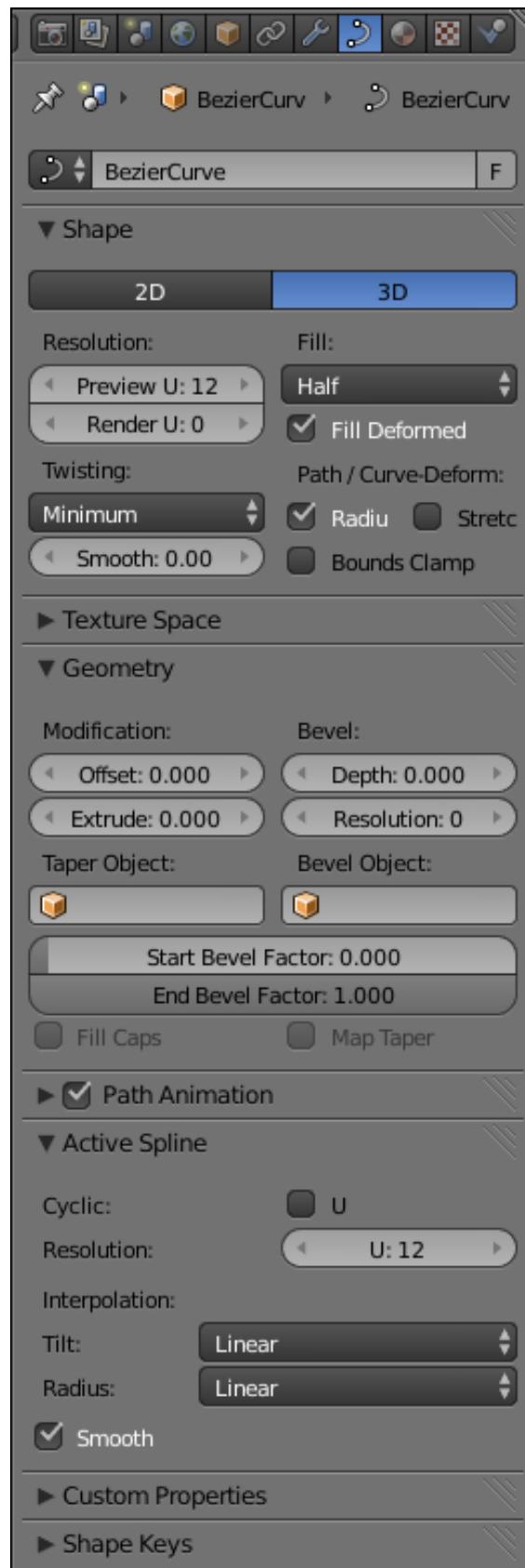
Now try increasing the “Resolution:” value, and you should see the V cross section start to smooth out, until at about a resolution of 2 or 3, it looks like a curved half pipe.

Now look further up, in the Shape panel. There should be a popup menu with the title “Fill:” above it; for a 3D curve, by default the item selected is “Half”. Try the “Front” and “Back” items, and you should see that these give you just halves of your half pipe.

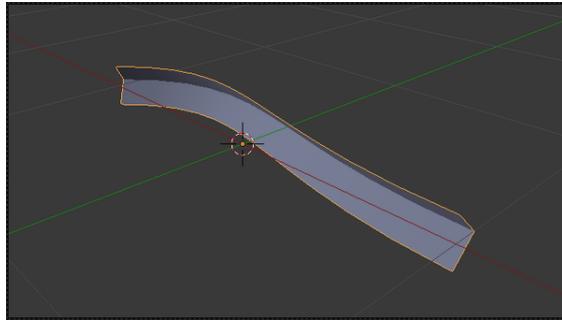
Now try “Full”, and your half pipe should now be a complete pipe.

You can also join the ends together by checking the “Cyclic: U” box under the Active Spline panel.

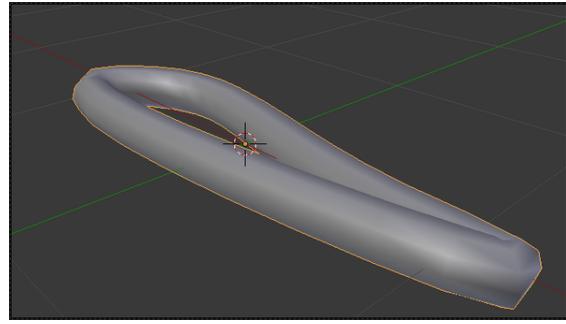
Start and End Bevel Factors: These reduce the extent of the bevel shape, so that instead of extending the full length of the curve, they go from and to the specified fractions of the length. These become more useful when the cross section of the shape is no longer uniform, when you apply a custom taper (below).



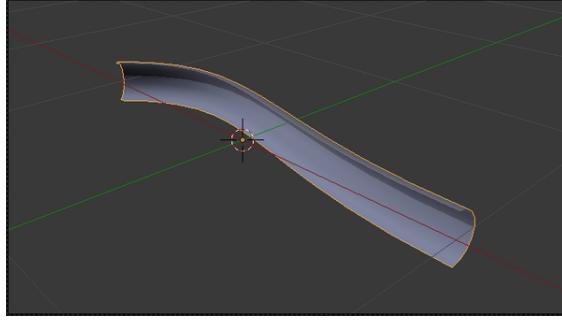
Now change the curve to 2D. The Fill options now become “Both”, “Front”, “Back” and “None”, where “Both” is like the “Half” setting for 3D curves. Note there is no



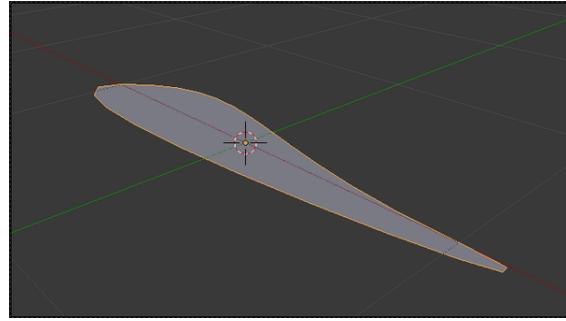
Bézier curve with bevel = 0.1, fill = half, resolution = 0



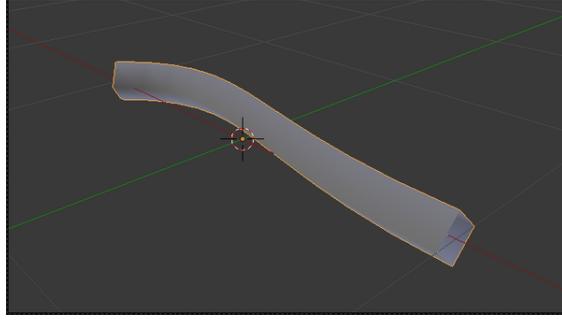
Bézier curve with bevel = 0.1, fill = full, resolution = 3, cyclic



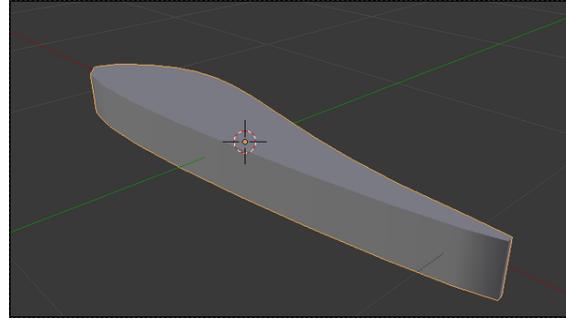
Bézier curve with bevel = 0.1, fill = half, resolution = 3



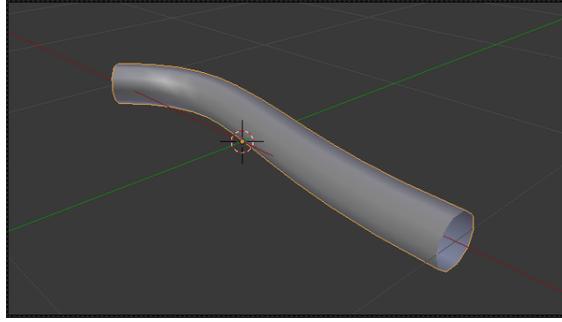
Bézier curve, 2D, cyclic



Bézier curve with bevel = 0.1, fill = full, resolution = 0



Bézier curve, 2D, cyclic, extrude = 0.1

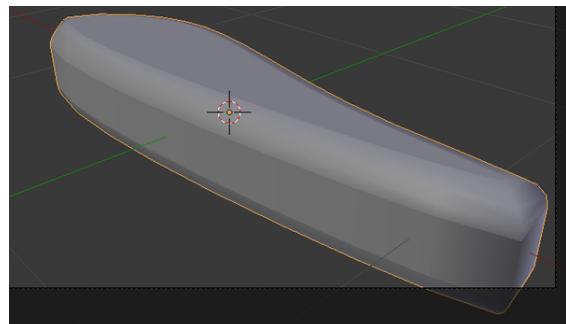


Bézier curve with bevel = 0.1, fill = full, resolution = 3

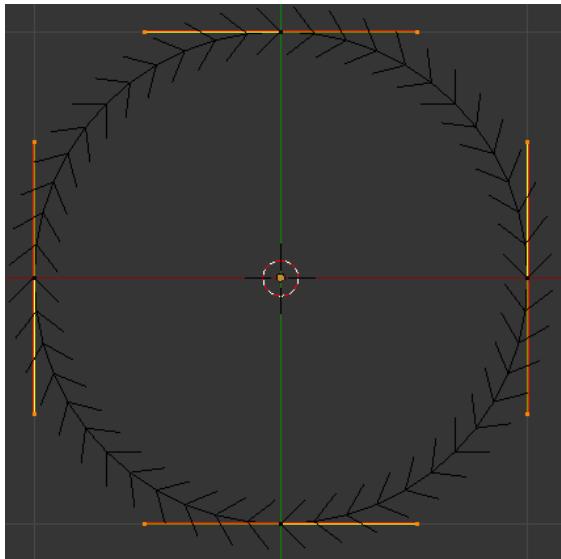
equivalent of “Full”. But there is a new feature: if you check the “Cyclic: U” box, it will fill in the entire interior of the curve!

Try this with a Bézier circle or a NURBS circle, and you should get a pancake-like object. For added flavour, give it a nonzero Extrude value, and this will make the shape

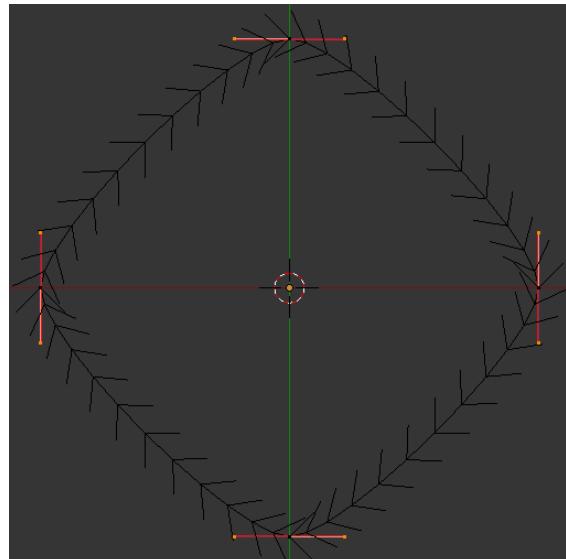
even thicker.



Bézier curve, 2D, cyclic, extrude = 0.1, bevel = 0.1, resolution = 3



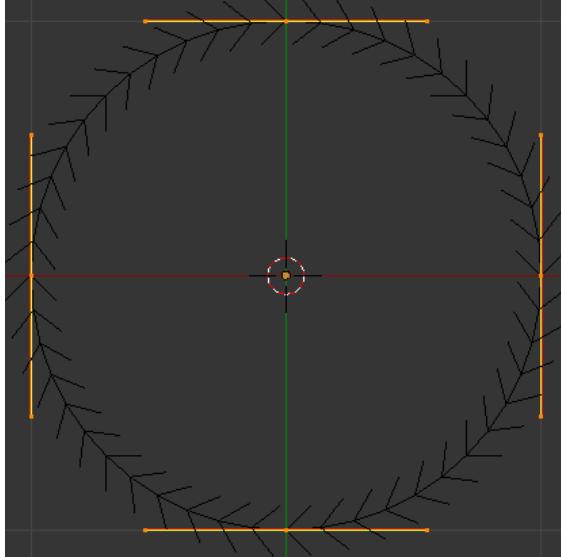
selecting just the end points of the handles



handles scaled to 0.4

2.50.2 Custom Bevel

Now we will try using another curve to supply the cross section. Add a Bézier circle shape. TAB into Edit mode. Now, select *just the end points of the handles (not the control points themselves)*. In the Pivot Point menu, select “Individual Origins”. Now S scale down to 0.4.

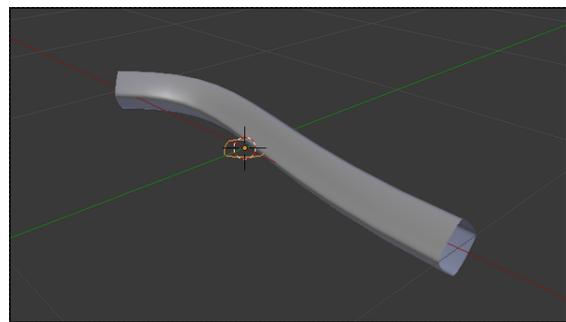


if they highlight like this, you’re doing it wrong

The result should be something close to a square, but with rounded corners.

Go back to your first curve, and find the field titled “Bevel Object:” in the Geometry panel in the Curve Context properties. Click on it, and a popup menu should appear, and you should see the name of the second curve (e.g. “BezierCircle”). Select that name, and you should see your bevelled curve immediately take on the cross-section

of the new curve.



In fact, you may find the bevel object is a bit large; select it and, while still in Object mode, S scale it by a factor of 0.1. The result should appear something like this.

Select the second curve again, go into Edit mode, and try messing around with the control points: you should see your changes immediately get reflected in the shape of the bevelled curve. You may find that the orientation of the second curve doesn’t coincide with the orientation of the bevel cross-section. This is easy to fix, because any change you make to the rotation and position of the second curve *in object mode* will have no effect on its use as a bevel, so you can freely reorient and reposition it to make it easier to match its shape up with the bevel cross section. (Scaling does, however, have an effect.)

2.50.3 Custom Taper

Now try adding a *third* curve object. (As with custom bevels, only curve objects will work.) Go back to your first curve, find the “Taper Object:” field (it should be next to the “Bevel Object:” field you’ve already used), and select the name of your newly-added curve. The bevelled

shape will now most likely squish down in a most peculiar way. Select your third curve, that you are using as a taper, and go into Edit mode. Now try adjust control points, and observe the effect on the bevelled shape: what you should find is that the object-local Y-coordinate of each control point governs the thickness of the bevelled shape at the corresponding position along the shape given by its X-coordinate, while the object-local Z-coordinate doesn't have any effect.

As with the bevel shape, you can freely rotate and reposition (and this time, even rescale) the taper shape *in object mode*, and it will have no effect on its taper function: only alterations of the control points in edit mode have an effect.

2.51 NURBS Patches

What Blender calls *surfaces* are more commonly referred to in computer graphics as *patches*. It makes sense to stick to the commonly-accepted terminology, particularly when talking with users of other software.

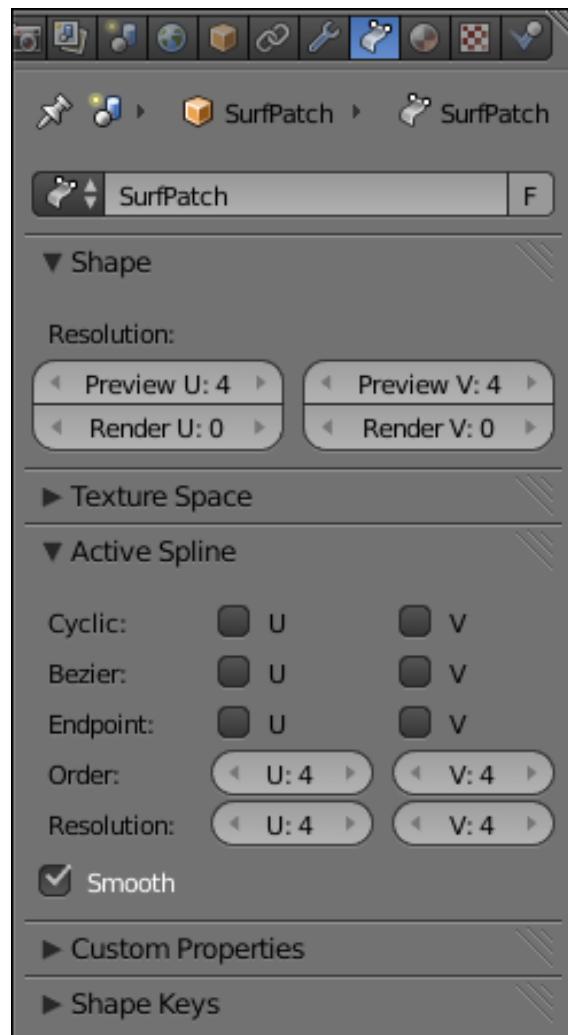
You previously saw how a NURBS curve consisted of a single row of any number of control points. A NURBS patch consists of an n -by- m grid of points, where n and m can be any positive integer (and not necessarily equal). The grid has a rectangular topology, but of course the points may be positioned anywhere in space, to shape the curve accordingly. The resulting object can look a bit like a mesh in edit mode, but it behaves very differently.

2.51.1 Your First NURBS Patch

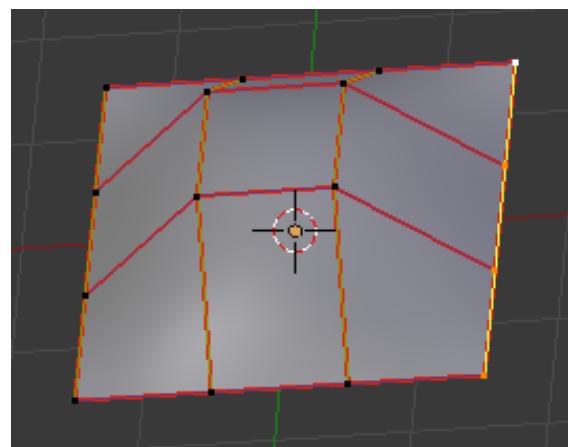
Start a new document, delete the default cube, bla-bla-bla, but stay in Object mode. Add a new NURBS surface: SHIFT + A → Surface→NURBS Surface. Switch to Edit mode, and find the Surface Context  in the Properties window.

For NURBS curves, you previously had the “Cyclic: U” and “Endpoint: U” options for making the curve open or closed, and extending all the way to the endpoints or not. Now you also have “Cyclic: V” and “Endpoint: V”, because the surface has two dimensions, and you can control these settings independently along each dimension. Try checking just one Cyclic box at a time, and the patch turns into a closed ribbon shape along the corresponding dimension; check both, and it forms a solid object, shaped perhaps reminiscent of a pillow or an unusual loaf of bread.

As with the curve case, you can move selected points around to alter the shape of the curve, and adjust the “W” value of each point to control how strongly it attracts the curve. Radius-scaling and tilt settings are still adjustable, but don't seem to achieve anything, since you can't apply a bevel to a surface.

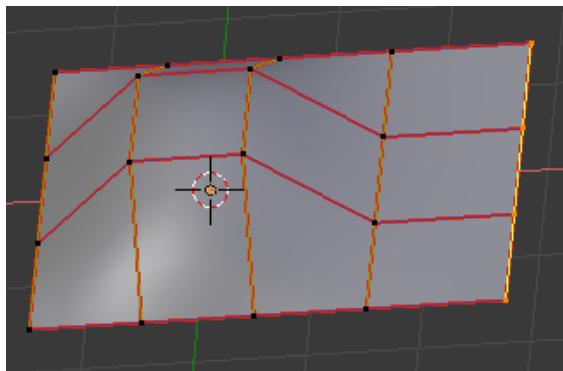


Adding and Removing Control Points

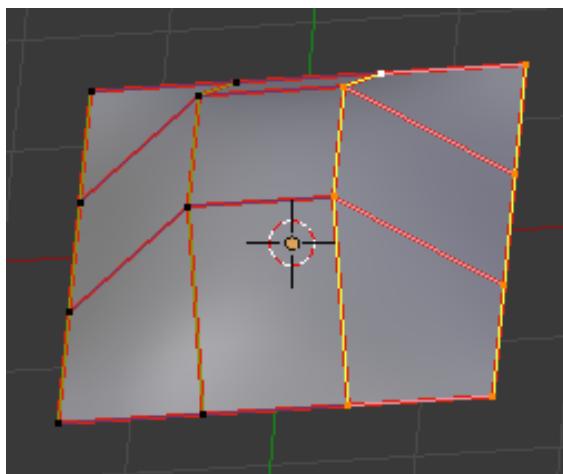


Select a complete row of points

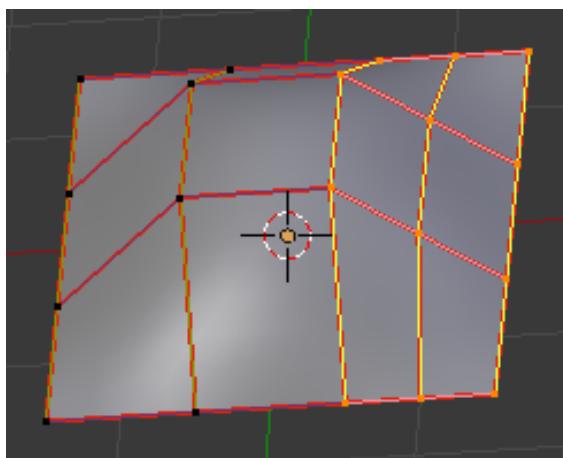
Adding and deleting points in a NURBS surface need to follow some rules. The irritating thing is, if you try to do things not in accordance with these rules, Blender will simply ignore you, with no error message.



Extrude the selected row of points



Select two complete rows of points



Subdivide by adding a new row of points

Remember how I said that the control points form a n -by- m rectangular grid? Any addition or deletion of points must preserve this characteristic. Thus, you can only add or remove points an entire row or column at a time. For example, to extend the patch, you select all the points along one outermost edge of it, and press E to add the same number of new points. Or you can select all the points in two adjacent rows or columns, and

use W → Subdivide to add a new row/column of points in-between. Similarly, you can only delete points by selecting an entire row or column of them at a time.

NURBS Curve, NURBS Circle?

The first two options in the SHIFT + A → Surface menu are “NURBS Curve” and “NURBS Circle”. Try adding these objects; at first glance, they look exactly like the “Nurbs Curve” and “Nurbs Circle” entries in the SHIFT + A → Curve menu. However, regardless of appearances, these really are surfaces, not curves.

To observe the difference, ensure all points are selected, and now use E to extend the curve: this will create a *whole row* of new control points, instead of just one! As with the different options in the SHIFT + A → Curve menu, the ones in SHIFT + A → Surface offer premade objects for you to choose whichever is the most convenient starting point for the shape you actually want to create.

2.52 Deforming Meshes using the Curve Modifier

2.52.1 Understanding the Curve Modifier

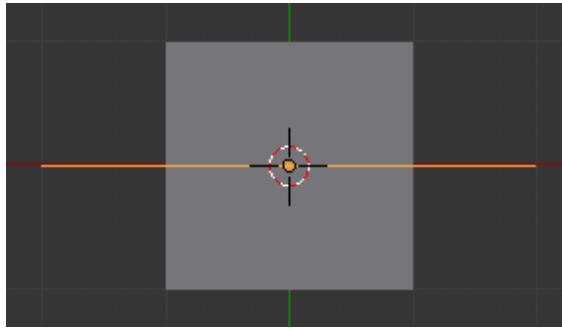
The *curve modifier* lets you use a curve shape to deform a mesh. The mesh will follow every twist and turn of the curve as far as it can bend, depending on how many vertices it has—clearly, the more vertices there are, the more faithfully it can follow the curve.

The curve modifier is also very easy to confuse yourself with if you’re not careful in how you set it up. A common situation is you try to move the deformed object in one direction, but it ends up moving in a completely different direction! This happens when you choose a deformation axis that doesn’t correspond to the predominant orientation of the points in the curve.

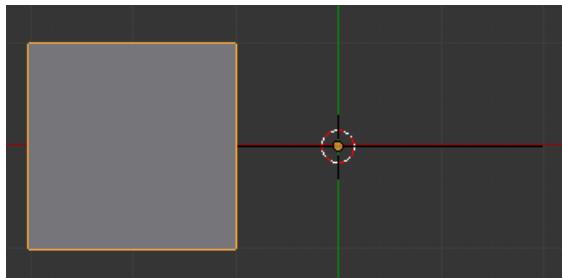
There are in fact some predictable rules that govern the behaviour of the curve modifier. Once you understand these rules, you shouldn’t need to keep trying different things at random to try to achieve the effect you want, you should be able to go straight for it.

A Simple Example

Let’s do an illustrative example to clarify things. Start a new Blender document; delete the default cube, and add a grid object instead, with the default 10×10 subdivision to give it plenty of vertices that can be deformed. Also add a path curve on top of it—this will give you a NURBS curve with 5 points, initially in a straight line. The result should look like at right.



Grid plus path objects, ready for action



Grid deformed by path, default settings

Now RMB on the grid, go to the Modifiers Context in the Properties window, and add a new Curve modifier which you will find in the “Deform” column. Click under “object” in the empty box and choose “NurbsPath”.

The grid object should immediately jump a little way in the direction of the negative X-axis. Why?

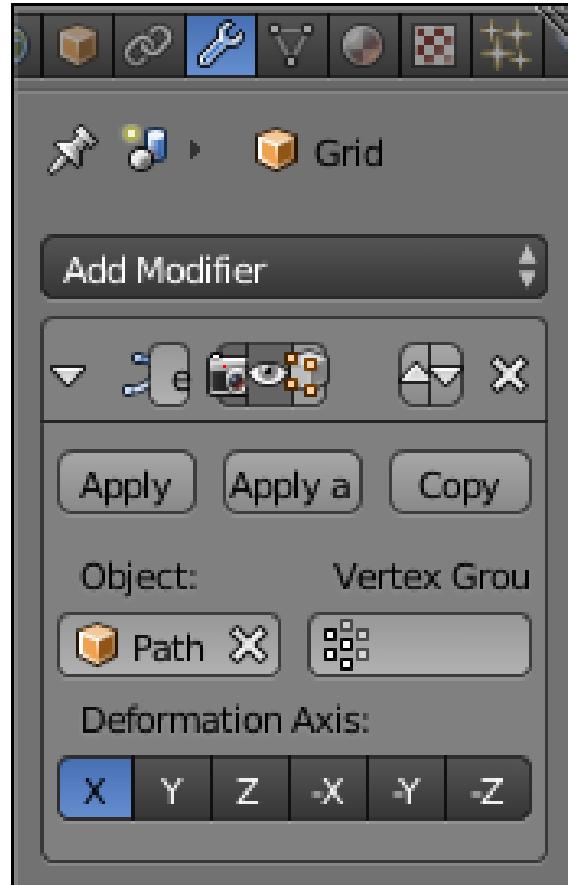
If you select the curve and TAB into Edit mode, you should see that, while the origin of the curve’s local coordinate system (the fat orange dot) is in the middle of the curve, the first point (the one at the end of curve in the same direction that the “centipede legs” are pointing) has a negative X-coordinate. Select just that point, and from the Snap menu SHIFT + S , do “Cursor to Selected”. Now TAB back into Object mode, and from Object→Transform do “Origin to 3D Cursor”. This will adjust the origin of the curve’s local coordinate system to that first point (keeping the curve itself in the same place), whereupon the grid object should jump back to its original location.

You can of course freely move around, add and delete the other points. Just be sure to keep that first point always at the origin of the curve.

Now look at the settings for the Curve modifier. Notice those six buttons under “Deformation Axis:”?

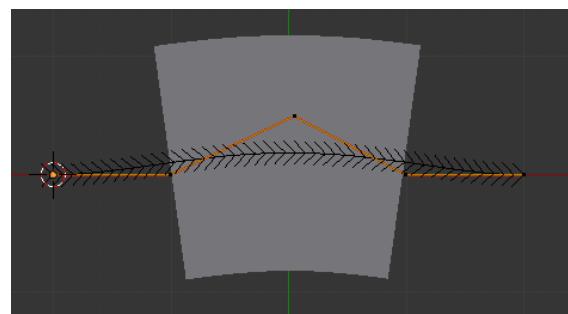
When a path curve is created, all its points initially lie along its local X-axis. Since the default for the Deformation Axis is “X”, all works pretty much as you expect. If the two do not coincide, then weird things happen when you try moving things around.

If you *really* want to mess around, you could go into Edit



mode on the curve, select all its points, rotate them to, say, orient them along the local Y-axis, then get out of Edit mode, select the deformed mesh, and change the Deformation Axis on its modifier to Y, and things should still work consistently. But why bother?

So long as you obey the above rules, you can move, rotate and scale the deforming curve *in object mode only* as much as you like, and the deformation will still behave in a reasonable fashion. Of course, you can do what you like to the deformed mesh in object or edit modes, and this will still be true.



Fiddling with the deforming path

Select the curve, TAB into Edit mode, select the middle point, and move it a little to one side, as at right. You should see the grid mesh immediately bend in a corre-

sponding way.

Now TAB into Object mode, select the mesh, and move G it around: notice how it tries to follow the shape of whatever part of the curve lies nearest to it. What shape does it take when you move it off an end of the curve? How about to one side?

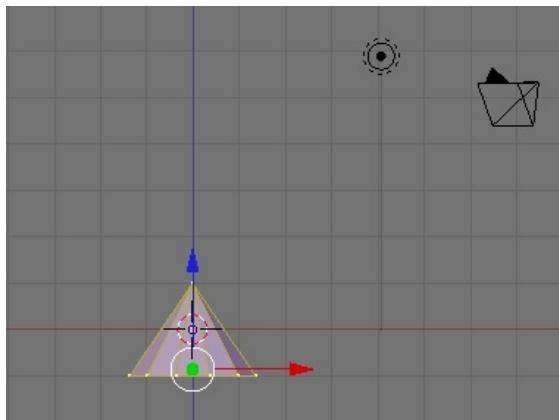
Alternatively, you can similarly drag the curve instead of the mesh, and the same thing will happen: it is only their relative positions that governs the actual deformation.

Also try changing the scaling radius ALT + S and tilt angle CTRL + T on particular curve points: note how the former causes the deformed mesh to get wider and narrower at those points, while the latter makes it tilt from side to side.

2.52.2 Another Example

Setting up your Mesh

While this example will be done with a cone primitive, you can use the default cube or another shape of your choosing so long as it can be loop subdivided along the axis you wish to curve. Delete the default cube if you're not going to use it as your base mesh, and add your chosen primitive. In this example, we'll be using a Cone.



A basic cone in Edit Mode

Do everything with the 3D cursor at the center.

- Set your view to the X-Y Axis (Top View) by pressing **NUM7**, the 7 on the keypad.
- Press **Shift+A > Mesh > Cone**
- A smaller number of vertices are needed, since this cone will become just the tip of the finished shape.
- Reduce the number of vertices to 12.
- Press **TAB** to enter Edit Mode, or choose Edit Mode from the bottom of the 3D viewport.

- Switch views to the Z-X Axis by pressing **NUM1** (1 on the keypad). Your screen should now look something like the picture at left.
- If your cone is selected (one or more vertices are yellow) press **AKEY** (Select/Deselect All) until they are deselected.
- Select the point of your Cone with the **RMB** and drag it upwards with the blue arrow. You can hold down the **CONTROL** key if you want to constrain the scaling to set units. For this example, 5 squares of height were added to the cone.

Subdividing the Cone

- Go to front view by pressing **NUM1**

The best way to do it is by selecting the side edges and subdivide them by pressing **WKEY** then clicking on *Subdivide*, after that you can select the number of cuts you want. You need at least “5” segments. The more segments you have, the smoother your curved cone will be. be sure you have “Limit selection to visible off” and to first deselect all “A key” and you are in “edge select” mode

- You can also do: still in Edit Mode, press the **NUM8** repeatedly to rotate your view to the underside of your cone. Select the center vertex of the circle, and hold down **CONTROL**, then press the **+KEY** on the keypad. This will select all of the adjacent vertices of the base. Then extrude using the **EKEY**. Pull the new vertices away from the cone a distance at the Z-axis and click the **LMB** to set them in place. Press the **SKEY** and widen the new base of the cone a bit.

Making a Curve

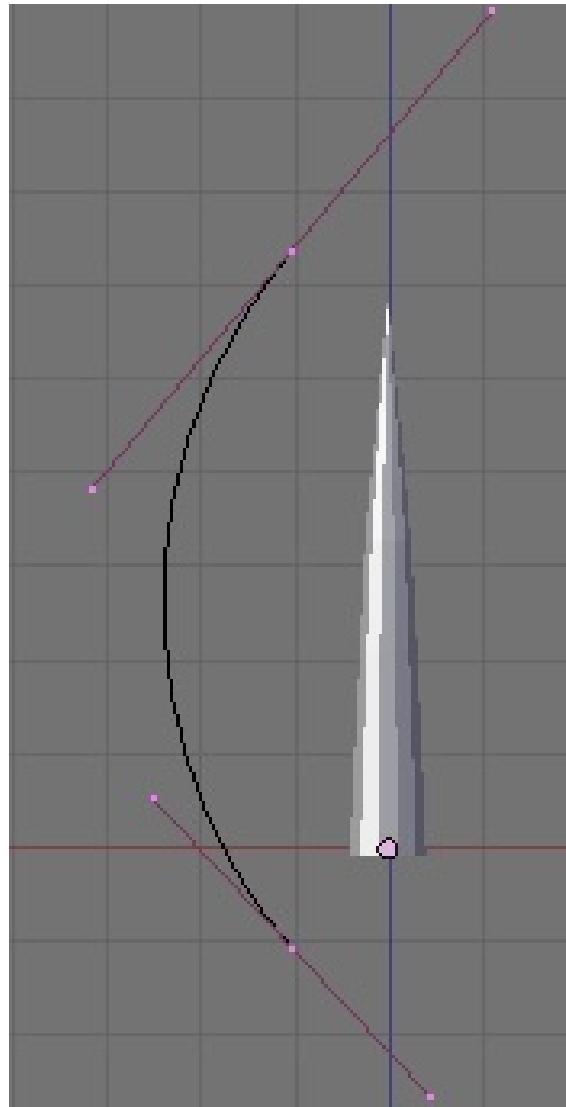
- In Object Mode, Z-X orientation ('**NUM3**), **make sure you have nothing selected and press Shift+A > Curve > Bezier Curve**
- Rotate your curve 90° (holding down the **CONTROL** key to make it rotate in 5° intervals) Note- or just R-KEY and then type 90, so that it lines up with your cone. Move it off to one side so that its not hidden by your cone.
- Go into “wireframe”
- Grab the center vertex of each end and adjust its rotation (**RKEY**) until you have a nice shallow dome shape.
- Scale the curve in size (**SKEY**) until it is larger than your cone.
- The name is highlighted in the “Outliner” Window: “BezierCurve”



Extruded cone

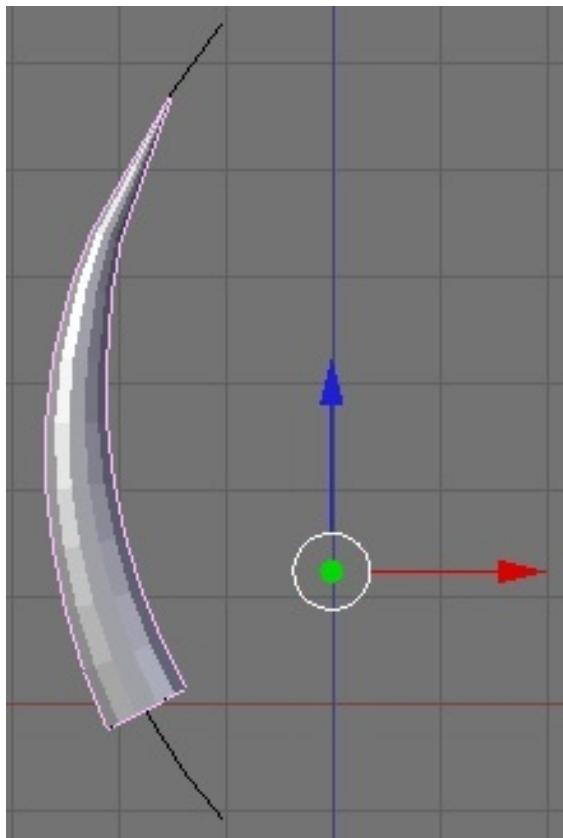
Applying the Curve to the Mesh

- Select the cone in Object Mode. Choose in the “Properties” Header the “modifiers” tab and click “Add Modifier”
- Click the *Add Modifier* button and choose *Curve*



from the popup list.

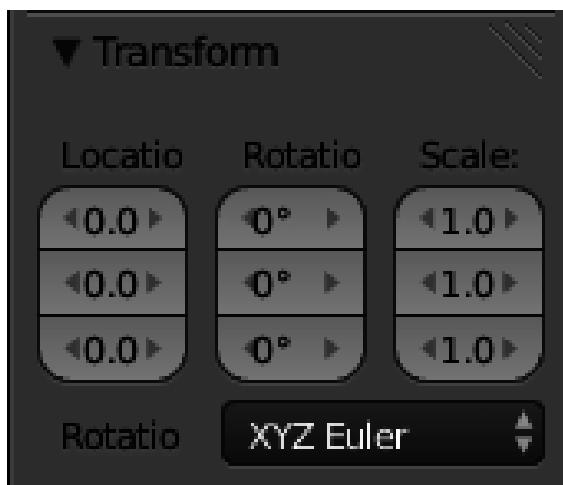
- Under Object tck the empty box, and select the name of the “Bézier Curve” (it will be “Bezier-Curve” without the quotations if you left it as the default.)
- Notice the six buttons underneath, They are **X**, **Y**, **Z**, **-X**, **-Y**, **-Z**. They affect which plane the curve deforms. For this example, you'll need to select the **X** button.
- Move your Cone so that it overlays the curve in the ‘3D View’ windows, and notice how it follows the Bézier Curve.
- You can modify the Bézier Curve as well as the cone repeatedly until you are happy with the design.
- If you switch to Edit Mode, notice that the cone returns to its straight orientation.



Modified cone

- To apply the deformation to the mesh permanently, in the Modifiers panel, click the *Apply* button next to your Curve modifier.
- Move the curve up 'till the cone is around the curve

2.53 The Empty Object



Every object in a scene has common settings in the Ob-

ject Context. This includes the common transform settings shown at right: location in the scene, overall rotation and overall scaling. Most objects also have additional properties, such as the geometry of a mesh and the materials that govern how it appears in the render.

The **Empty object** has none of these additional properties. It has the overall transform settings, and not much else. And it has no appearance in the final render. So what is the point of having such a thing?

In fact, it has many uses, such as:

- In the Physics Context, it can be set as the source of a force field (e.g. wind).
- It can be used as the **parent** for multiple other objects. That way they can be moved together just by moving the Empty, rather than each child object individually.
- It can be used as a target object in an **array modifier**, where it adds its transformation to the copies of the object being modified by the array.
- It is a handy marker and place holder if you are moving things around. You can mark an object's original position, and then put it back easily where the empty object is.

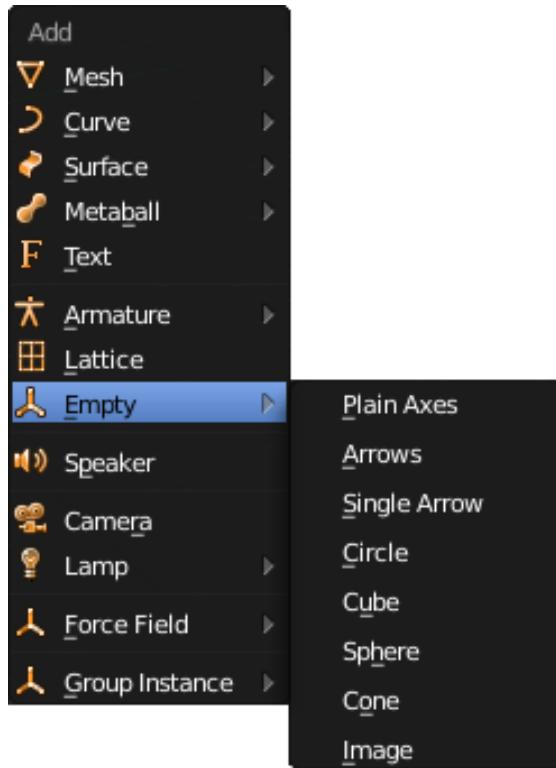
In these situations, the invisibility of the Empty in the render is an advantage, because it can be placed wherever necessary, without introducing unwanted clutter into the final image.

2.53.1 Using an Empty With the Array Modifier

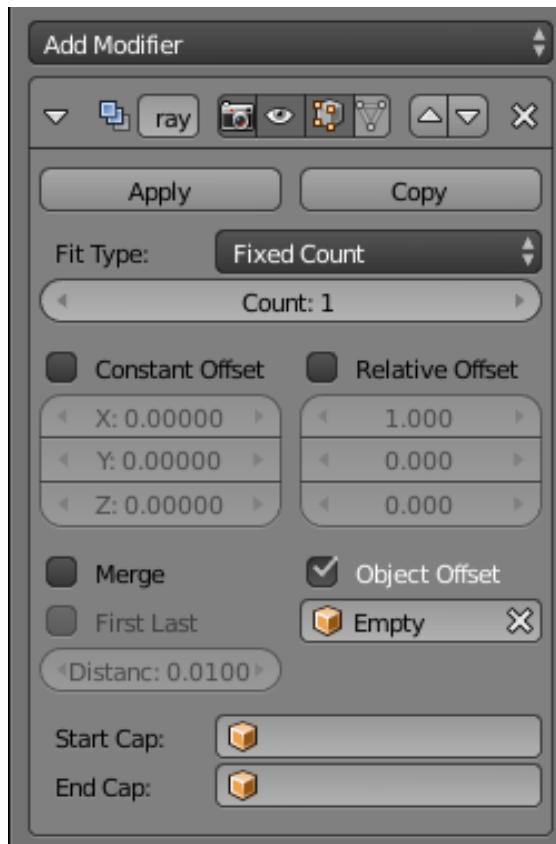
Open a new default Blender document, complete with default cube. Add a new Empty object to the scene. It doesn't matter what type of Empty you choose; the options just control its appearance in the 3D view, not the actual behaviour of the Empty. Besides, you can change your mind about this later anyway.

Now select the default cube, go to the Modifiers Context and add an Array modifier. In the properties header, there are three different ways the modifier can produce its copies of the original object: via a Constant Offset, Relative Offset, or Object Offset. The Constant and Relative offsets only allow a simple displacement for the copies, but the Object Offset looks at the full transformation of the specified object, which can include rotation and scaling.

Here I have unchecked the default Relative Offset, checked Object Offset, and from the popup menu for selecting the target object, I have selected my Empty (which by default would have been called "Empty").



The types of Empties



But if you look at your 3D view, nothing seems to have happened: the cube is still sitting, unmoved, in its original position. In fact, it might be hard to see the Empty, because it would have been inserted in the same position as the cube, so it ends up within it. If you switch to Wireframe Z view, you should be able to see the Empty sitting within the cube.

Before doing anything further, increase the Count field in the cube's array modifier to 3. This will give you a clearer idea of the effect of subsequent manipulations. Doing so will not have any noticeable effect to start with, because all the copies will be lying right on top of each other.

Now select the Empty, and move it to one side: you will immediately see a second copy of the cube follow the empty, and the third copy move *by the same amount in the same direction*, so it ends up beyond that.

Return the Empty to its original location, and this time try moving the cube. As the original cube moves, you will see the second copy stay with the Empty at the centre of the scene, while the third copy moves beyond that, by the same distance and in the same direction as the second copy is removed from the original cube.

Now if you return the cube to the same location as the Empty, select both and try moving them together, you will see no additional copies of the cube appear, because they remain exactly on top of the original cube.

The rule for using an Object Offset with the Array modifier is this:

In other words:

- the translation applied is the difference between their origins.
- the rotation applied is the difference between their object rotations.
- the scaling applied is the *ratio* between their object scaling factors.

Try moving the Empty to one side, as before; this time leave it there, so the three copies of the cube are nicely spread out. Now try applying a rotation to the Empty; see how this applies the corresponding rotation to the second copy (the one located where the Empty is), while the third copy gets *twice* that rotation.

Return the Empty to its unrotated state, and this time try rotating the cube: see how the second copy (where the Empty is) stays unmoved, while the third copy gets the opposite rotation.

Undo the rotations, and now try scaling: shrinking/enlarging the Empty correspondingly shrinks/enlarges the second copy, while the third copy gets transformed by the *square* of the shrink/enlarge ratio (the ratio multiplied by itself, or alternatively the ratio raised to the second power). Or shrink/enlarge the original cube, and see the second copy stay unchanged (as before), while the third copy enlarges/shrinks by the *inverse* ratio.

When you change the transformation of the original cube, the second copy always stays unchanged because the Object Offset transformation is the *inverse* of the transformation you are applying to the original; hence it always cancels out for the second copy, but you can see corresponding increasing powers of it applied to subsequent copies.

Editing the Modified Object

Make sure the Empty is positioned so the array instances are well separated. Select the cube, TAB into Edit mode, select all the vertices, and try transforming them: moving, rotating or scaling.

Note the difference in behaviour from doing this to the cube in Object mode: this time all the copies of the cube transform along in unison. This is because we are not affecting the object transformation, which is what controls the behaviour of the array modifier.

Arranging Object Copies in a Circle

Which brings us to a convenient trick for a common need: arranging copies of an object in a neat circle.

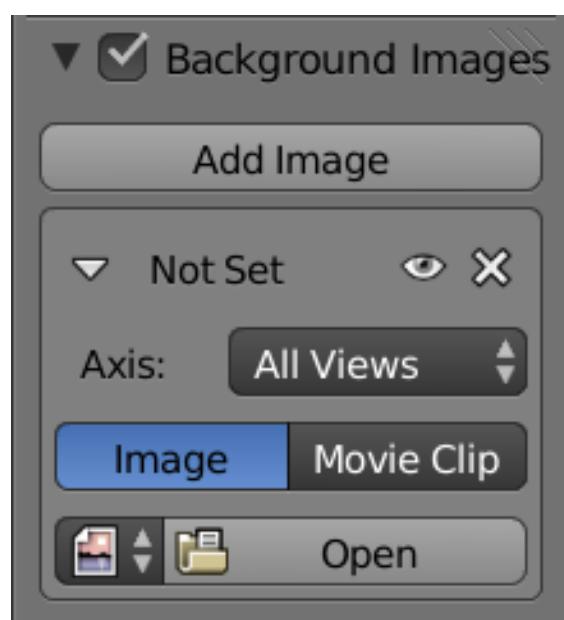
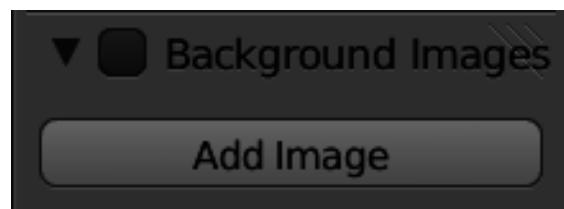
The easiest way to do this is start with the cube and the Empty in their original central positions. Suppose you want 8 copies of the cube in your circle; the angle between them then needs to be $360^\circ \div 8 = 45^\circ$. So rotate the Empty by 45° . Now select the cube. Set the Fixed Count field of its Array modifier to 8. TAB into Edit mode, and move all the vertices to one side. You should see the other 7 copies of the cube correspondingly move away from the centre in different directions, maintaining a nice neat circle arrangement. Your cube mesh is simply being rotated about its origin, which is not where the vertices are, but where the Empty is!

2.54 Background Images

You will often need to use reference photos to guide your modeling. Among the many things you can do with photos and other images in Blender, it is possible to use them just as guides in the 3D view, such that they do not appear in the final render. Furthermore these images will only be visible when the view is *orthographic* and exactly aligned to one of the X, Y or Z axes (or the camera view). This way, they can be used as a precise reference for positioning elements of the model.

In the 3D view, make sure the Properties Shelf N is visible. Look for the Background Images panel; it will most likely be collapsed, so expand it. Initially it will look like at right, with no background images in your view.

Check the box at the top. Click the “Add Image” button once, and this will add one entry to the list of background



images, as at right. Initially there is no image opened in this entry; you will need to click the “Open” button and select an image file.

Once you have chosen an image file, the list entry expands with additional options.

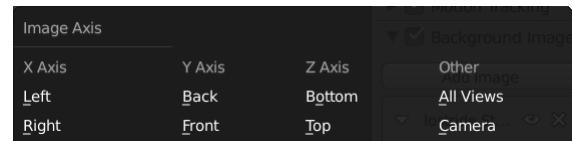
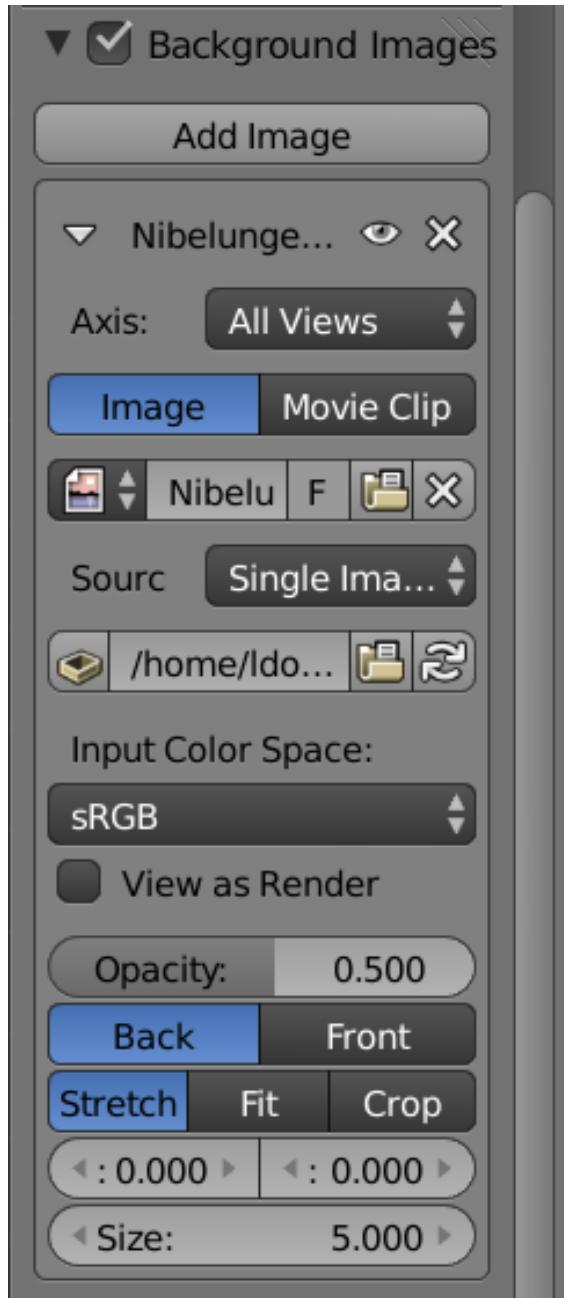
Clicking on the eye icon lets you temporarily hide a reference image, without actually removing it from the document. Or you can uncheck the box at the top of the Background Images panel to temporarily hide all reference images; just check it to make them visible again. Click the X to delete an entry from the list of background images.

If you have multiple scenes in your document, the document's background image settings apply to all of them.

If you don't have an image handy to experiment with, how about downloading this simple 2D one from Wikimedia Commons. Flat images with no perspective work best for use as a precise modelling reference.

Having loaded your image, it might not immediately appear in the 3D view. For it to show, you must be in a *perfect* (axis-aligned) view (i.e. one of NUM1 , NUM3 , NUM7 , CTRL + NUM1 , CTRL + NUM3 or CTRL + NUM7) which is in orthographic, not perspective, mode (use NUM5 to toggle between these), or alternatively it will show in NUM0 the camera view.

You may not want the same reference image visible in all



menu lets you choose which view this image is visible in.

2.54.1 Other Ways To Bring In Reference Images

Maybe you don't want to use a reference image for precise, axis-aligned modelling, but only as a rough guide. There is a standard Blender addon called "Import Images As Planes", which will create a plane object and apply an imported image to it as a texture. This can then be aligned whichever way you want in the scene.

2.55 Aligning Vertices with a Guide Image

Note: Some pictures are outdated.

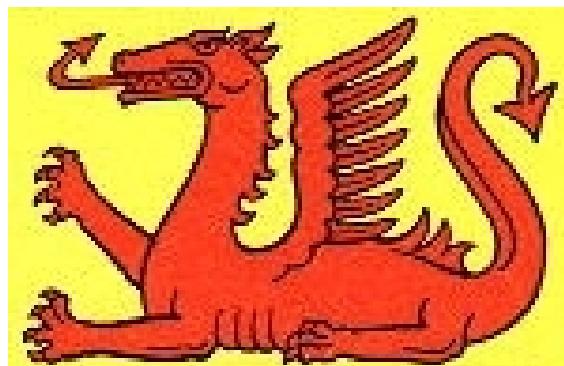
This tutorial is about using guide images to place vertices in their proper places in 3D space. The second tutorial is on how to take good reference pictures. This tutorial assumes that you have completed all previous tutorials.

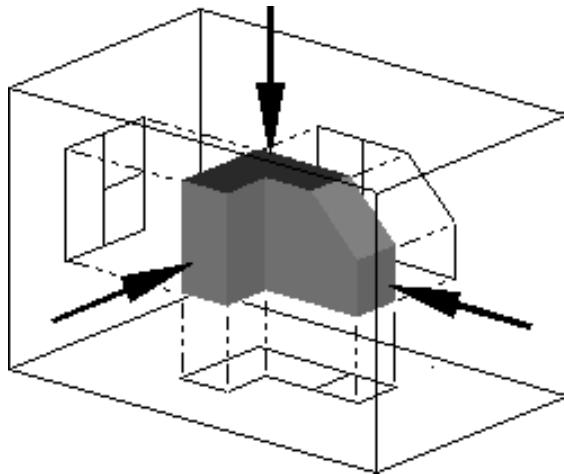
This tutorial describes the use of the *background image* feature of Blender to assist in creating models of 3D objects. The background image provides a reference for the dimensions of the object, similar to the way the floor plan and elevation views of a house provide dimension information for the actual house. Guide images are not rendered, and may be removed after the model is completed, or they may be retained as part of the internal documentation of the model. Background images are not generally useful for other "image" purposes such as materials, textures, and actual background images in a scene, just as a floor plan is not actually visible after a house is constructed.

2.55.1 Background: orthographic projection

orthographic projection is a technique used by architects and engineers to describe a three-dimensional object by the use of several two-dimensional images, or "projections." In architecture, the projection as seen from above is called the "plan view" or "floor plan," the projection as seen from the front is called the "front elevation," and the projection as seen from the right side is called the "right elevation." Orthographic projections are intended to assist builders in creating an actual 3D object.

views. For example, you might have one reference image for top view, another for a side view, and so on. The Axis





orthographic projections in six directions

A Blender user also wants to create a 3D object, in this case a model. Blender provides a way to use a set of one or more orthographic projections as a guide for object creation. Because the images are conceptually “behind” the object from the appropriate point of view, Blender calls these 2D images *background images*.

Each background image is located “at infinity” in an orthographic view, and there may be one image in each direction: back, front, top, bottom, left, and right. we can use one or more background images to assist our modeling effort.

For example, to model a house, we can put the “floor plan” on the bottom, and build the house above it. We put the front elevation image on the back background, and build the house in front of it. We put the right elevation image on the left background, and build the house to the right of it. The background images can be diagrams such as floor plans, or they can be photographs of an object taken from sufficiently far away to provide undistorted dimensional information.

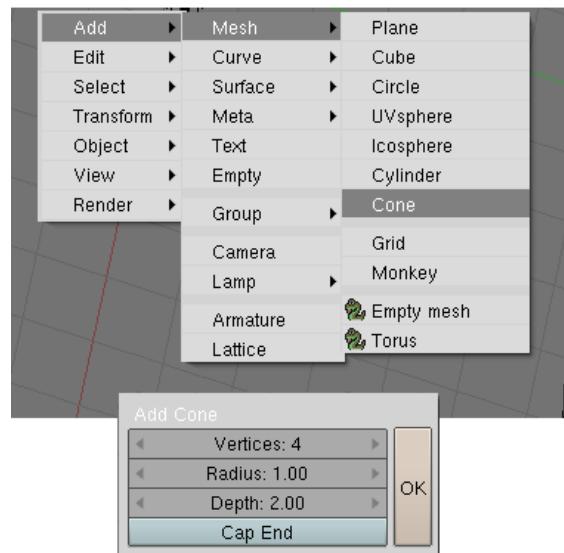
Reasonably enough, an orthographic projection is only useful in Blender when in orthographic mode. Blender enforces this to prevent you from making mistakes: the orthographic image is only displayed in the background when you are in the orthographic view mode.

2.55.2 Making a Simple Pyramid

First we are going to create a pyramid the easy way. Then we are going to show how to use different viewpoints and images as a guide to place vertices correctly in 3D space.

Get rid of the default cube. Press Shift+A and select Mesh→Cone. Set the number of vertices to 4, and Set “Capp fill Type” to “Nothing”. Click OK. There’s your pyramid.

Note: Do not render this looking up from the bottom, as it will appear invisible, as the interior faces of models are



ignored by most rendering engines.

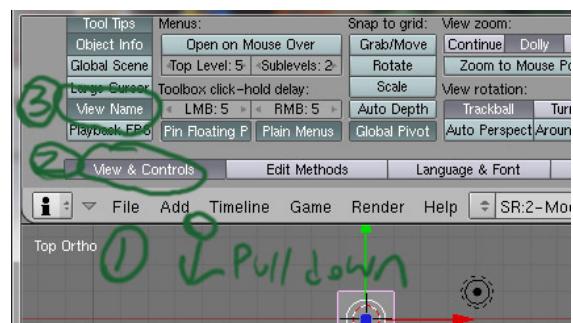
Save the top vertex for later steps

Enter Edit mode (TAB). First, unselect all the vertices by pressing A . Next, select the bottom four vertices of the pyramid and delete them with DEL or X . The only vertex left will be the vertex which makes the tip of the pyramid. This will be used later.

2.55.3 Using the guide images

Now that we have the pyramid the easy way, let’s learn how to use guide images as references to build models.

Window Layout

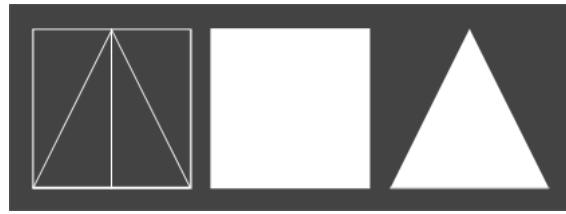


Split the Main 3D view window in to 4 windows (2 x 2).

Reminder: to split windows, move the mouse to the border of the view, when the cursor transforms into arrow, right-click and choose

“Split Area”. (Explained in the guide: [Noob to Pro/Blender Windowing System](#).)

The point of view in each window are like this:



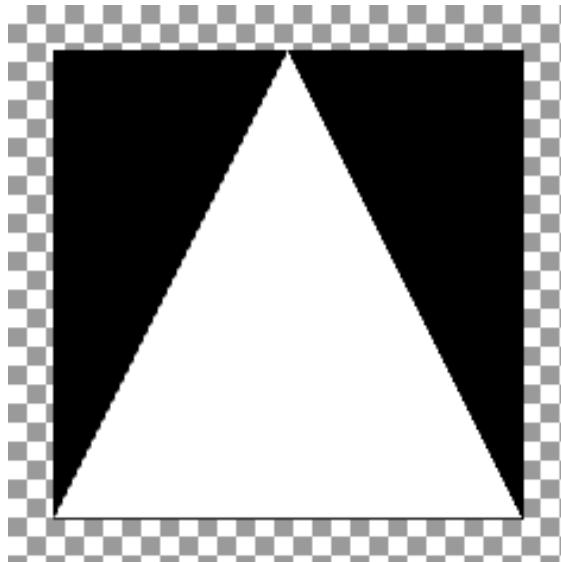
And if you click on View, you can see that these windows are respectively:

By knowing which view you are looking at you may find that you quickly get the idea of what you are doing and can proceed somewhat intuitively in this section on your own without following all of the step by step instructions.

Note: The quick way to achieve this layout is to go into Quad-View (CTRL-ALT-Q)

Guide images

Now, we need some images. These come from a source outside of blender. For this tutorial, we need a floor plan of our pyramid, a front elevation view, and a side elevation view. The floor plan of our pyramid is a square, and the front elevation view is a triangle, as is the side elevation. We will cheat and use the same source image for both the front and side elevation. We can make a picture of a white square and of a white triangle in the GIMP, Paint :, or some other image editor. Or we can find appropriate images somewhere.



Save the triangle with a black background

Method 1 download and use Download the black and white triangle image on the right of the screen and use that. This image is, minus the checker pattern at the border, 198 x 198. (click it once to get the larger version, right click on the larger one, and save)

The square and triangle relationship

Method 2 roll your own You want to make a triangle, for the ground plane you don't need a picture.

Noob note Make sure that the *drawing* of the square is *square* and not just rectangular. Make the triangle the same width and height as the square. Make sure the apex of the triangle is directly above the midpoint of its baseline.

Specific instructions for Photoshop Make a square selection of “n by n” size, remember the value of “n”. Fill it with white color and save. To create a triangle of needed properties make a rectangular selection of same (n by n) size, on a new layer, click **RMB** on your document, choose “Transform selection” option in the pop-up menu. Once you are in “Transform selection” mode, right-click the blank image again. This time the pop-up menu would be different. Choose “Perspective” from it, and with **LMB** drag one of the two top vertices toward the other. Once the vertices meet (in the top-center of the image), exit the transformation mode, and fill the resulting triangular selection with white.

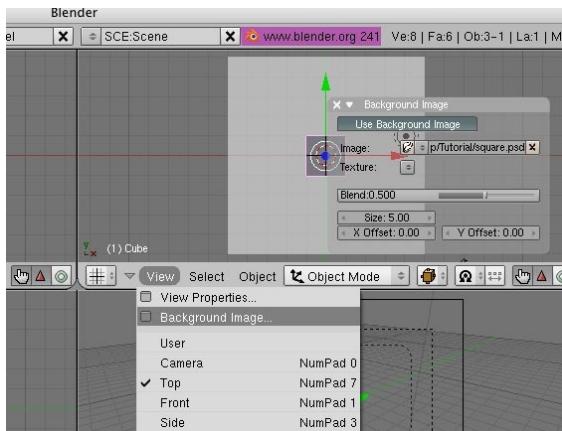
Save the files to a place that is easy to access. Blender only supports the TGA, PNG, and JPG image formats.

Specific instructions for Gimp Turn on the grid (View->Show Grid, View->Snap to Grid), use the rectangle select with a fixed aspect ratio of 1:1 (in the tool options panel) to select a square that you can flood fill. For the triangle, use the node tool to draw a triangular path, convert to selection (Select->From Path) and fill it. Or you could just use Inkscape...

Save the files to a place that is easy to access. Blender only supports the TGA, PNG, and JPG image formats.

Background Images

Load the images (as described in the previous module) like this: In the 3D view, make sure the Properties Shelf (**N**) is visible. Look for the Background Images panel; it will most likely be collapsed, so expand it. Initially it will have no background images to view. Check the box at the top. Click the “Add Image” button once, and this will add one entry to the list of background images. Then, with ‘Image’ selected, use the Open button to navigate to your image file.

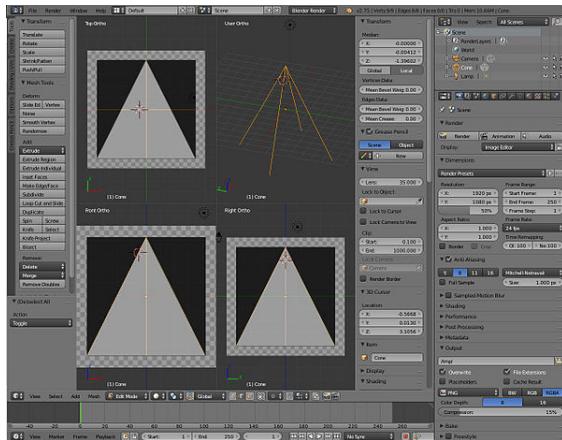


Load the white triangle (the front elevation view): this places it infinitely far “behind” the model. You can see the image in the front view. Similarly, load the right elevation (coincidentally, you can see the same triangle image in the right view window, to place it infinitely far to the left).

If necessary, zoom out so that you can see the whole background image in each view.

Now you have now placed your guide images for making your pyramid.

Sides



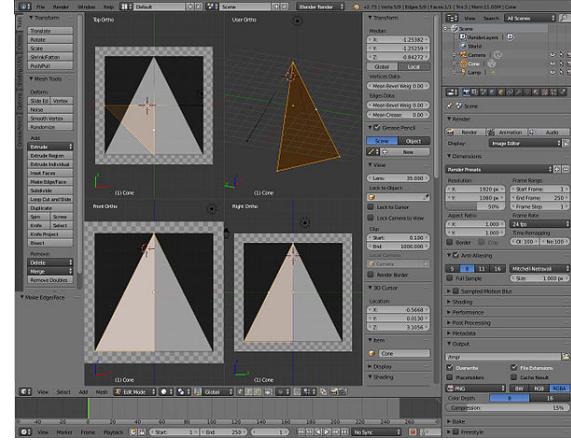
Make sure you're in Toggle Quad Mode. The vertex that is left will be the topmost point of the pyramid. Use the **GKEY** to move the vertex around. To get it in the right spot, line it up at the top most point in the **front** and **right** windows. If you look in the **top** window the vertex should appear to be in the center. Make sure to keep the vertex highlighted for the next step.

Now you can begin to create individual vertices with **CTRL + LMB**. Be sure you don't create vertices in the Top view cause that will confuse you completely.

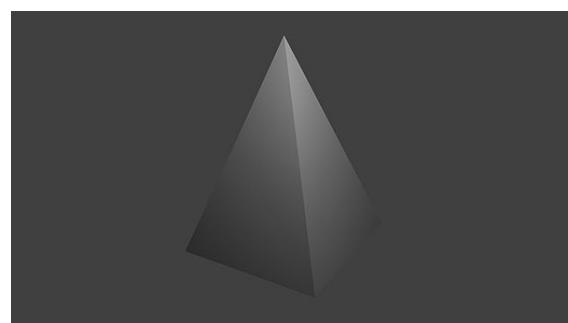
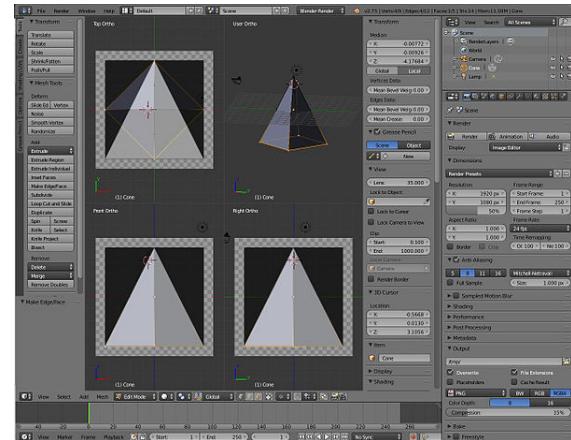
First line in the pyramid in front view then place a new edge between the last made vertex and the top vertex by

selecting them and hit “F”. Then do the right side, after selecting the top vertex, exactly the same way. (Don't do the bottom side)

Make Faces



Select the side vertices of the corners were you want a Face and Tick “F” Do this for every Face you want to make.



2.56 Modeling a Fox from Guide Images

This tutorial assumes that you have completed all previous tutorials.

2.56.1 Method 1

Get the pictures of the model

Tip: The images here do not line up. Some need to be rotated and others do not match in size. They will be kept so that you can get the “real feel” for this project.

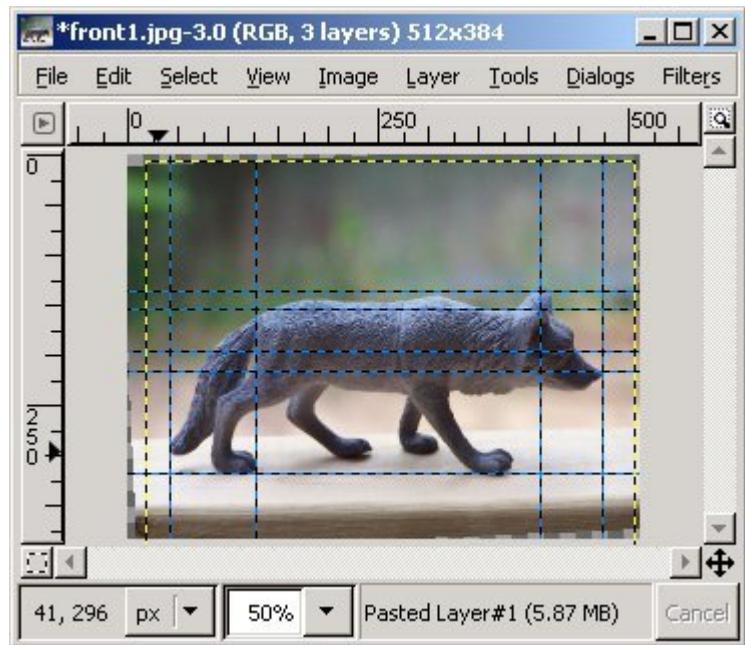
If you have a puppy and a digital camera, take three pictures of the cute little rascal and upload them. If you don't have a puppy, any object or small animal will do. Ideally, the photos will be looking straight down at the top of the puppy, a side view, and a front view. It's important that the puppy be in the same pose in all three photos! Or at least close to the same pose...we all know puppies don't stand still very long.

You could use two mirrors. One is placed next to the puppy at 45 degrees to the camera and 45 degrees to the puppy. Another is placed above the puppy, also at 45 degrees to the camera and 45 degrees to the puppy. This produces three images, one of the puppy (**front \ NUM1**), one of its reflection seen 90 degrees to the right (**side \ right \ NUM3**), and one of its reflection seen from overhead (**top \ NUM7**). Take the photo from a long distance away with a zoom lens to get close to an orthographic projection.

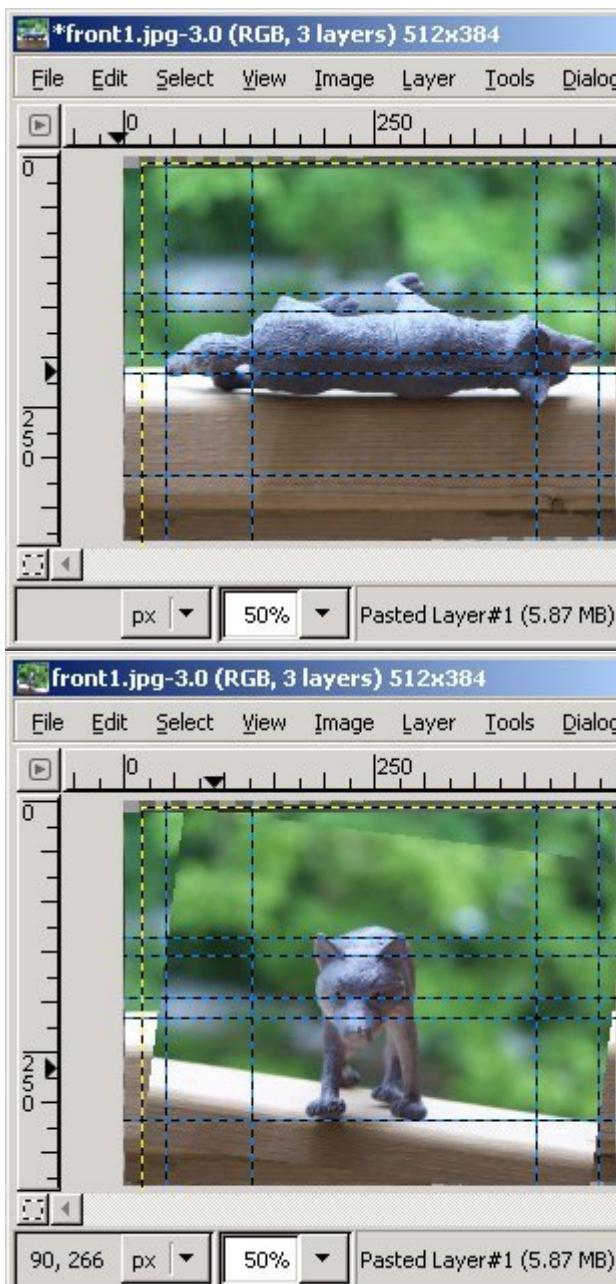
Or how about pictures of a toy wolf taken from 6 view points?:

- Left view
- Right view
- Front view
- Back view
- Bottom view
- Top view

Using your favorite image editor, such as PhotoShop or the **GIMP** (see detailed GIMP instructions below: Detailed steps to align images using **GIMP**), down-scale the images need to a reasonable size (I made mine 512x384), and then match them to each other. To match them, draw construction lines (pulled from the rulers above and to the left) on the left view for example to pick out key features. I picked the tail, the front of the back foot, eye level, tip of the ear, and the front of the nose:

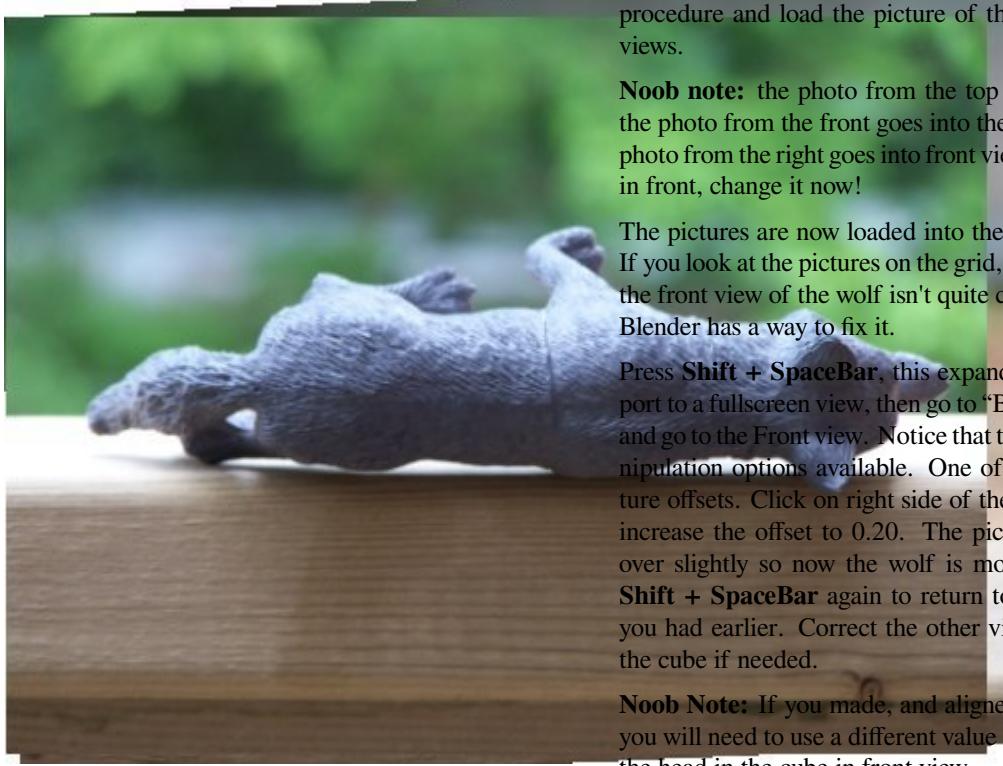


I found when I picked out these features that this first image needed to be rotated slightly. That completed, I proceeded to scale, rotate and shift the other two views (top and front) until they matched fairly well as layers on top:



Once I had the proper results I saved the resulting images, and these are the ones we will use in Blender.

The results are the files you'll need for Step Two:



Just right-click and save them some place where you can find them to load them into Blender for Step Two. You may notice the photos aren't perfect, but we'll use them just to show how you should deal with your real photos. When you are creating your own pictures to import, note parallax. In this example, parallax is present, and we'll attempt to compensate.

Get the Picture into Blender

Getting the image into blender is the easy part. The more difficult part will be creating the mesh, but first things first. Create a new file (*File → New*) to see the familiar default objects. Don't bother deleting the cube, we'll end up using it in the tutorial. Just as was done in the “[Making A Pyramid](#)” section, split the 3D Viewer into four views with **CTRL-ALT-Q**.

Each window will show you different XYZ coordinates.

Go to *View > Properties*, or press **N**. A new toolbar will open to the right of the viewport and by scrolling through you should see *Background Images*. Use the “Add images” bar and options will open up. Then check the box to display your images.

Click the *Add Image* button and several more buttons will appear. Now “Open” button. A new full-viewport window will appear. Explore this window a bit and end up selecting the image file of the wolf from the top view. What you should get is the picture of the toy wolf from above with the default cube on top of it.

Now load the top view of the wolf, click on “Open” select the picture and set “Axis:” to “Front”. Repeat the

procedure and load the picture of the wolf to the other views.

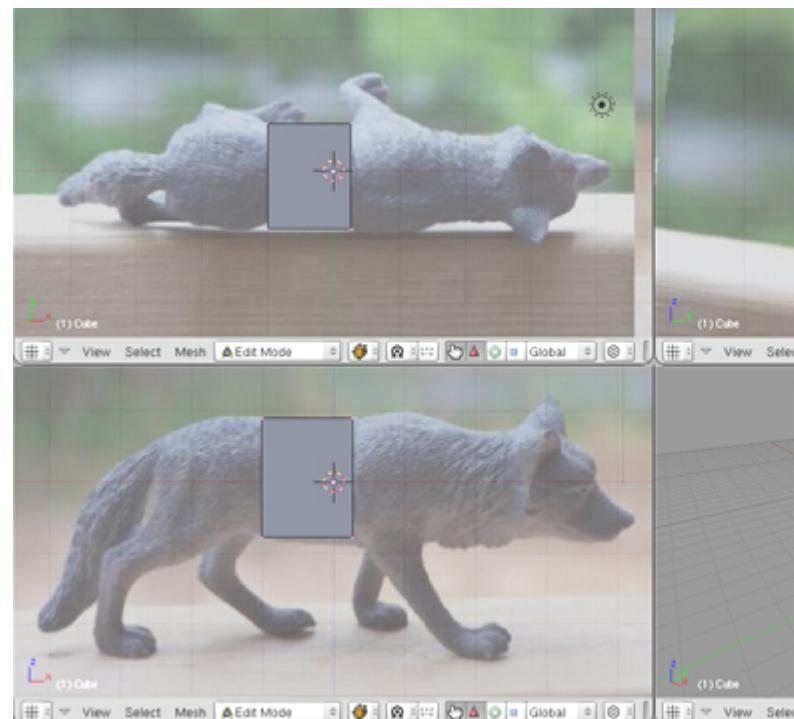
Noob note: the photo from the top goes into top view, the photo from the front goes into the right view, and the photo from the right goes into front view, if you have front in front, change it now!

The pictures are now loaded into the Blender viewports. If you look at the pictures on the grid, you may notice that the front view of the wolf isn't quite center. That is okay, Blender has a way to fix it.

Press **Shift + SpaceBar**, this expands the current viewport to a fullscreen view, then go to “Background Images” and go to the Front view. Notice that there are picture manipulation options available. One of these includes picture offsets. Click on right side of the ‘X Offset: 0.00’ to increase the offset to 0.20. The picture will be shifted over slightly so now the wolf is more centered. Press **Shift + SpaceBar** again to return to the window-setup you had earlier. Correct the other views also, and scale the cube if needed.

Noob Note: If you made, and aligned your own photos, you will need to use a different value than '0.20' to center the head in the cube in front view.

The setup work is now done! Let's start on actually making the wolf model.



Create a Rough Model

This is a brute force model creation using techniques discussed previously in this book. This section is meant to help you explore and become more comfortable with them. Do *not* try to follow the example to the tee. Your

wolf and my wolf will probably not look the same since you may want to add more or have less detail.

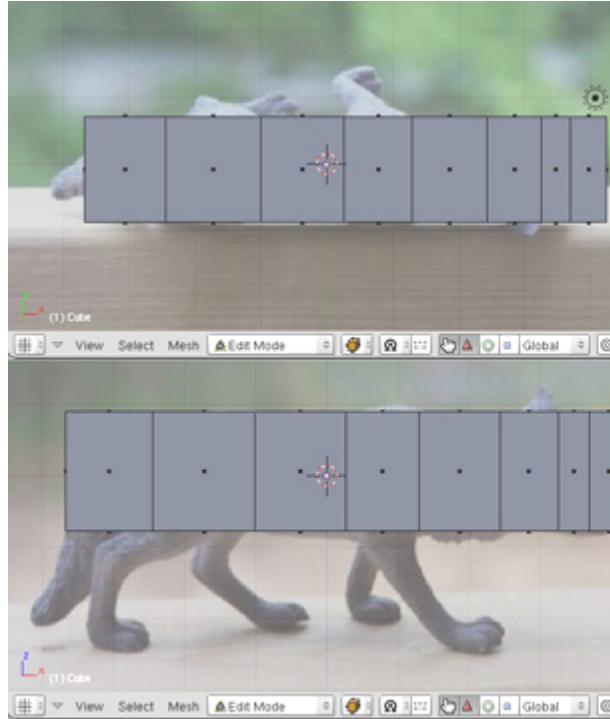
Noob Note: If you are just getting started with blender, this step may likely take several hours to complete, since you must use your own creativity to position things in 3D space, using 2D views. Just like sculpting, drawing, or oil painting, it will be extremely frustrating at first, but once you get used to the way it works, “modeling” will be much easier.

The rough fit stage requires either some planning or on-the-spot decisions. Think about where the wolf will have parts of its body flex or require parts jutting out.

The first step is to create a blocky wolf. Start out with a column of blocks using the extrude face command (select face, **EKEY**). Don't worry about snapping the vertices to the grid since we are working with an organic figure.

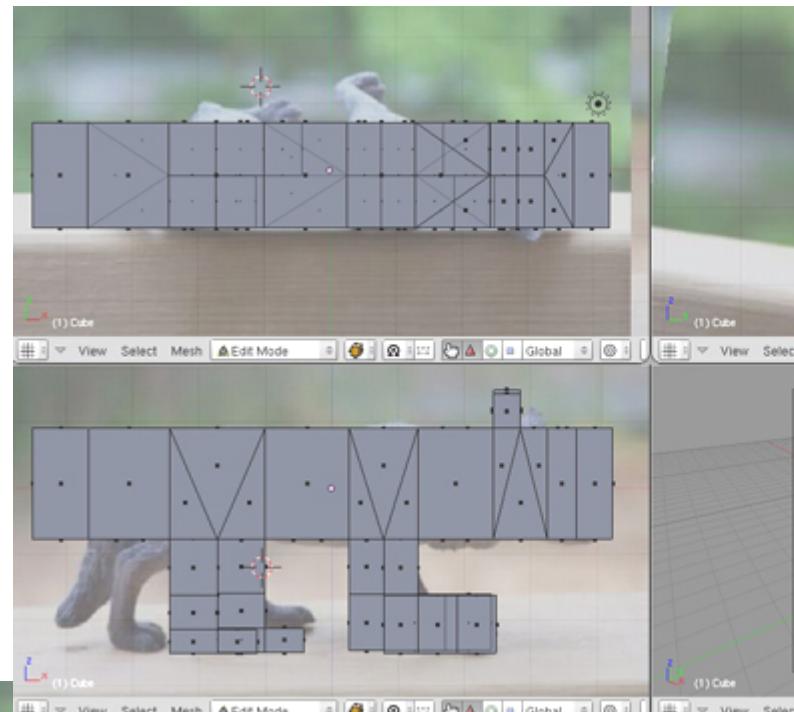
Noob Note: It is handy to do this in wireframe view, to see better how the rough model fits out

Figure 2.3.1 Body column formation



The next step is to split the ears and legs off of the body. Do this by subdividing (With “Quad/Tri Mode” checked) the appropriate faces. Save often, and if you make a mistake, go ahead and use the undo option (**CTRL ZKEY**). Also, if you find yourself looking at redundant faces, combine them (**FKEY**).

Figure 2.3.2 Appendage formation



If you are having trouble with this, try mousing over the perspective window (the one you designated with NUM0) and using the MMB to rotate the view so that you are looking at the underside of the wolf. Click on the face underneath the wolf that is alongside his front legs (use the side view to check this). We are going to subdivide this face in order to grow legs off of the new faces. To subdivide, press the **WKEY** and choose subdivide. You will see that the face has been divided into four. Take one of these faces and extrude it as many times as is necessary to make the right leg. Then do the same again for his left leg.

Noob Note: To avoid making the extruded faces share common vertices (and be connected to each other), either extrude the legs separately (as stated above) or select *Individual faces* in the Extrude pop-up menu. The same works for the ears.

Doing the ears is similar, except instead of working underneath you will start with the face on top of the wolf which is directly over the ears. Select this face and subdivide it once. Deselect everything using **AKEY**, then select one of these four faces and extrude it upwards once to make an ear. Do the same for the face alongside it to make the other ear.

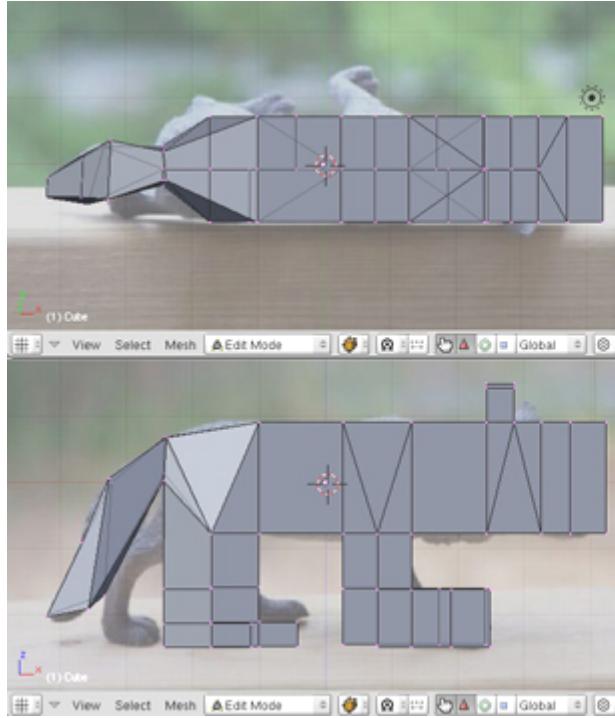
Finally, extrude the tail end of the wolf one more time, so that your wolf has as many divisions as the picture above.

Refine the rough model

Let's start refining the model starting with the tail. Try putting your viewports in wireframe mode by pushing **Z**, it may make things much easier. Line up the vertices over the wolf in each viewport by lasso selecting multiple ver-

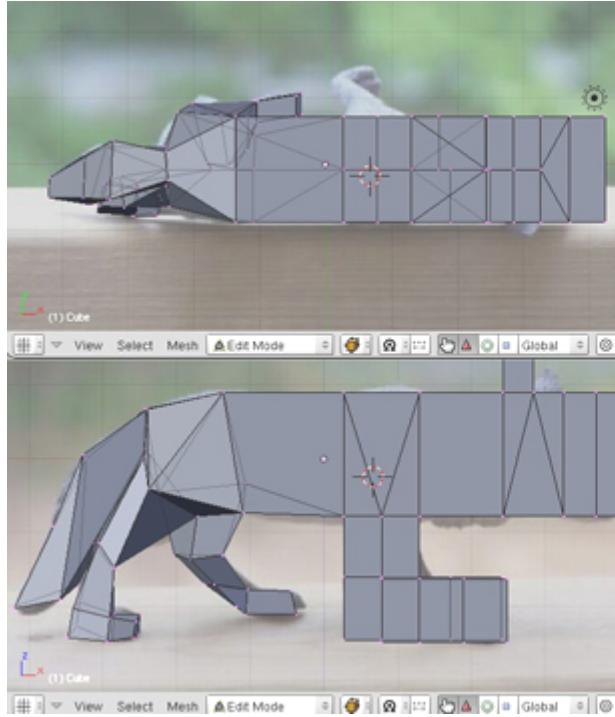
tices (**CTRL LMB, Drag**), then move to the right location with grab (**GKEY**).

Figure 2.3.3 Working on the tail



Continue onto the hind legs of the wolf. It is trickier to manipulate the legs so keep rotating a viewport to look at the model from multiple perspectives. Remember that we are working in three dimensions.

Figure 2.3.4 Working on the hind legs

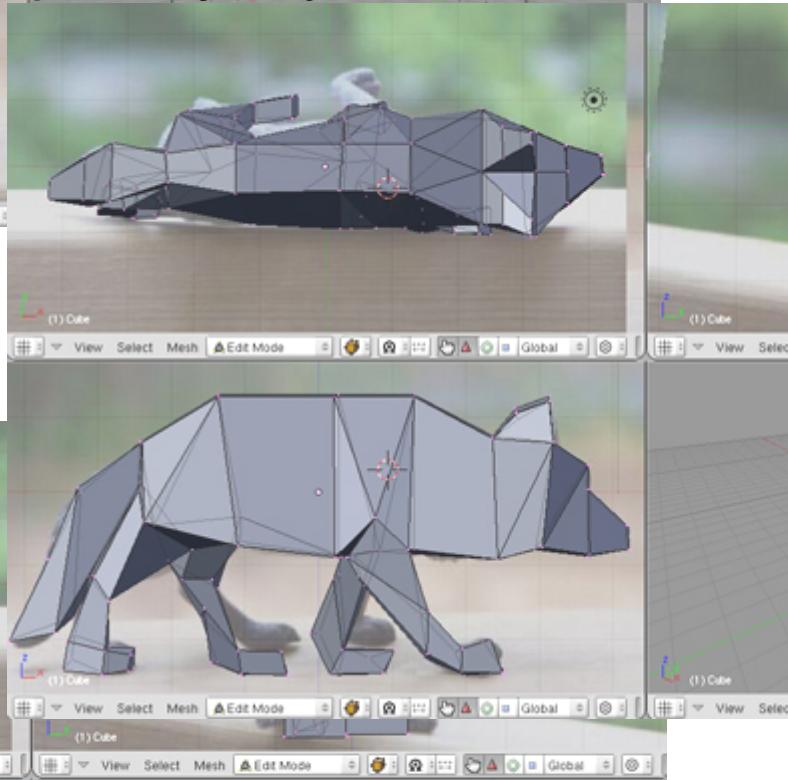


Continue working up along the wolf fitting the blocks to the pictures. If you have problems seeing the picture because the model is in the way, let's hide the model. In Edit

Mode, select the entire model by **AKEY** or by pressing **LKEY** when you have the cursor over the model. Simply pressing **HKEY** will hide the selected items. To unhide the view, use **ALT+HKEY**. By hiding and unhiding the model, or parts of the model, you should be able to keep using the picture as a guide. *Note: It is much easier to just switch layers by, let's say, pressing **2KEY** (not the Num-Block one) to hide the entire model (thus getting a look at the picture) and **1KEY** to reveal the mesh again.*

Once you have the first pass done, you'll notice that the model just won't fit all three pictures correctly. This is due to parallax. The most obvious example is the side view. The four feet should be level, as they are all standing on a flat surface. Since they are not, we'll just ignore some of the aspects of each picture and continue with the model. (This is a helpful example to show what you need to consider when taking your own pictures.)

Figure 2.3.5 Completed rough fit



Subsurf the model

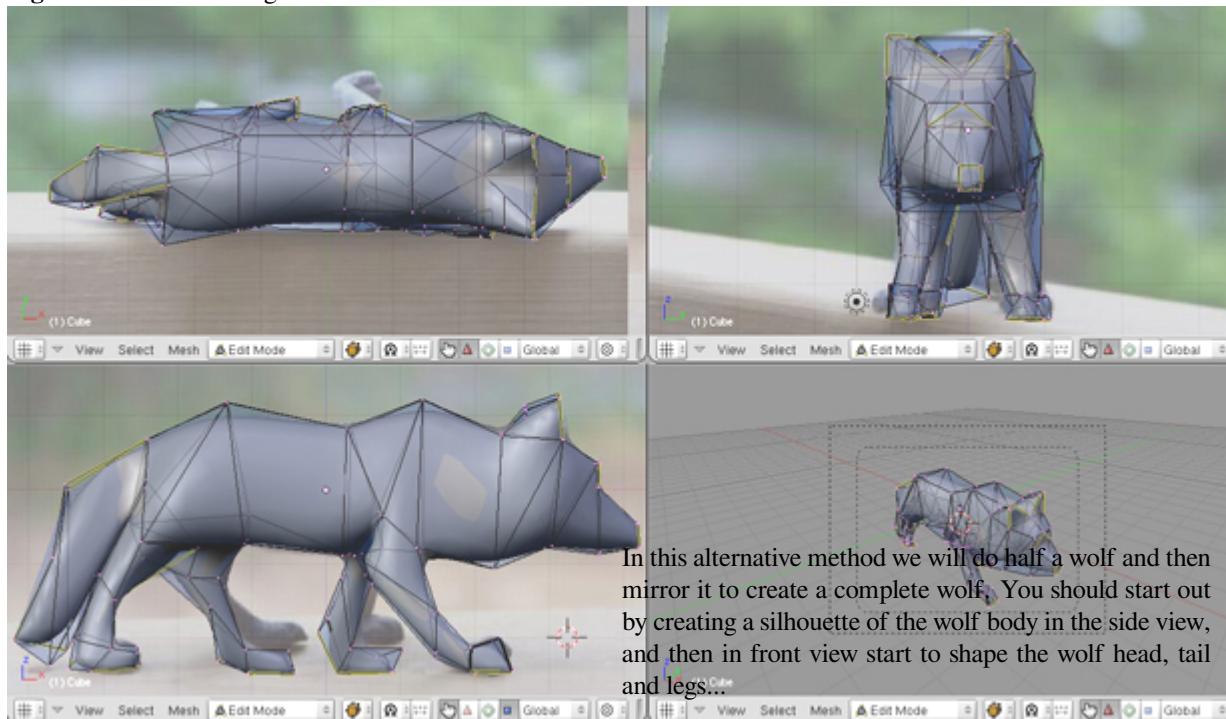
Now that the rough fit is done, let's smooth out the wolf. Add a Subsurf modifier and set the Levels to 2. The wolf will now be smoothed, but we want to add some of the hard lines back into the model. This may be accomplished with creased edges.

Select Edges or faces you want to crease and press **SHIFT + EKEY**. Use the mouse and pull away from the center until the Crease value is close to what you want. A value of **+1.000** will give you the sharpest look and is useful for places such as the bottoms of the paws. When an edge has been creased, the edge will be highlighted in yellow (positive crease) or black (negative crease). These highlights are shown due to the 'Draw Creases' button being

turned on.

In this example, I creased edges along the paws, tail, ears, and nose to give them some sharpness.

Figure 2.4.1 Creased edges

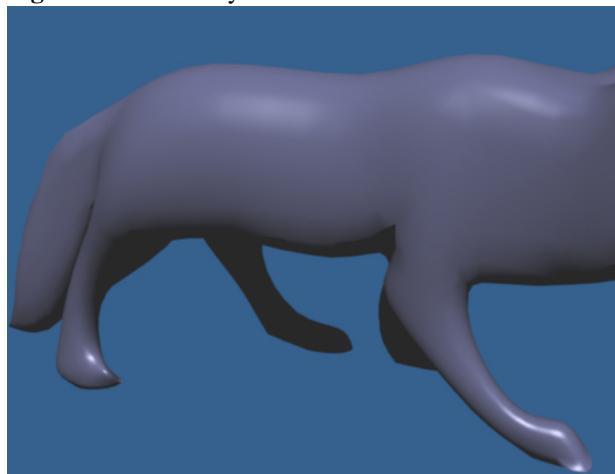


The last step is to refit the model to the pictures. You may have noticed that when the model was smoothed, the result didn't quite fit to the pictures. Now is a great time to tweak the vertices to fit to the pictures or add to/modify the model.

Noob note: if you see a weird edge on the body after smoothing, check the face normals (F9 > Mesh Tools More > Draw Normals; and then W > Flip Normals on the culprit faces).

And here is my basic wolf based on three pictures!

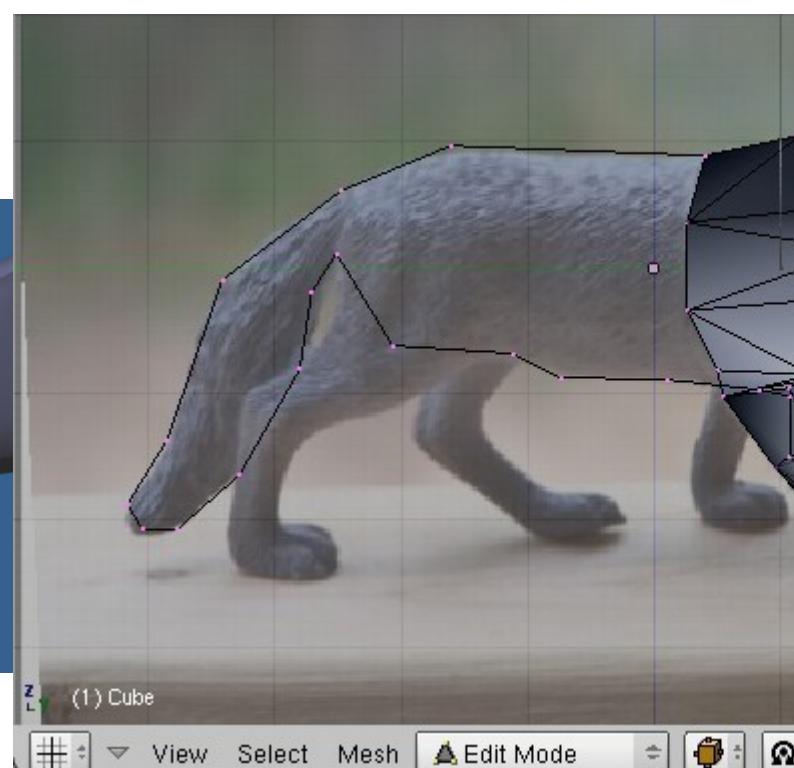
Figure 2.4.2 Final toy wolf model

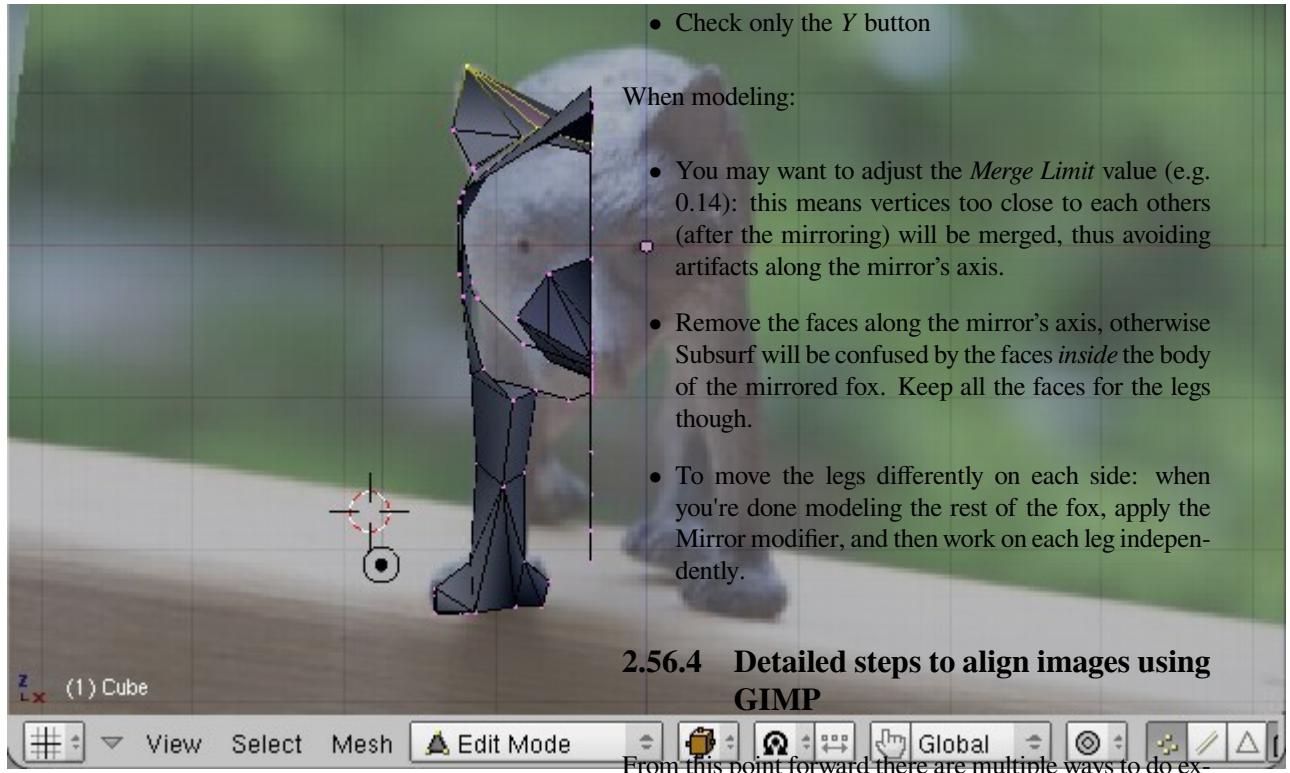


2.56.2 Method 2 (using mirroring)

In this alternative method we will do half a wolf and then mirror it to create a complete wolf. You should start out by creating a silhouette of the wolf body in the side view, and then in front view start to shape the wolf head, tail and legs...

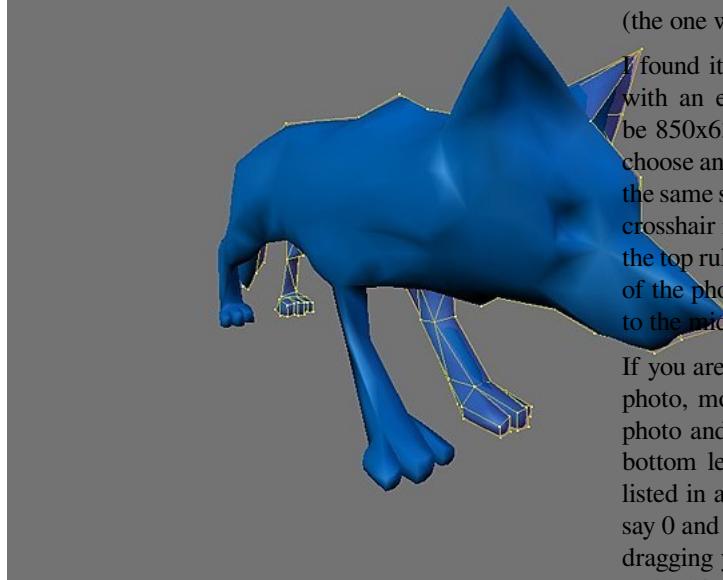
Noob Note: If you are attempting this method, it is best to avoid using triangles in rounded areas of a Mesh, because they tend to cause artifacts in your model (artifacts are protruding edges or other things that don't look realistic). Using quads is a much better alternative. Triangles should only be used on flat areas of your mesh if they cannot be avoided.





2.56.4 Detailed steps to align images using GIMP

Here is what you should end up with



2.56.3 Using mirroring with method 1

You can also use mirroring with the first method (the one with cube extrusions).

This can speed up your work as you don't have to do the same changes on each side of the fox - just once.

To start the mirroring:

- Go to Editing (**F9**)
- In the *Modifiers* panel, add a *Mirror*

From this point forward there are multiple ways to do exactly the same thing, however for simplicity's sake and so that I can be more detailed I will be using one method (the one which I use) and be using GIMP.

I found it best to size all the photos to a known width, with an easy way to find center. (Mine happened to be 850x638 pixels, I don't recommend that but you can choose any size you want really, as long as all of them are the same size). Then drag the construction lines to form a crosshair in the middle of the photo. To do this, click on the top ruler, and drag down to the middle (Exact middle) of the photo, then click on the side ruler and drag across to the middle (Again exact middle) of the photo.

If you are having trouble finding the exact middle of the photo, move the cursor to the very bottom left of your photo and the height of your photo will be listed at the bottom left of the GIMP interface. The numbers are listed in an (x,y) format so you want the first number to say 0 and the second to be the largest you can make it by dragging your cursor. The second number is the height, and half of that is the middle of your photo. You can do the same with the top ruler to find the vertical middle of your photo. Only this time the co-ordinates at the bottom left of the GIMP interface should list the second number (y) as 0, and the first number should be as large as you can make it by moving your cursor (to the upper right of the photo). Once you have your width again half of that will be the middle of your photo.

Then using construction lines put one at the top of your object, and the bottom of your object. Find the "height" of your object by the distance between them. Remove the construction lines from the top and the bottom, and place a new construction line above the horizontal center line by the half of the "height"(of your object). Now place a

construction line on both sides of your object and find the “width” (distance between the new vertical lines), then remove those construction lines and place a new construction line vertically half of the “width”(of your object) to the right of the vertical center line. Now cut the object out, and drag it so that the point you used as the “top” is on the horizontal construction line that is above the middle. Then Drag the photo left or right until the right edge of the object is on the vertical construction line you put in right of the middle construction line.

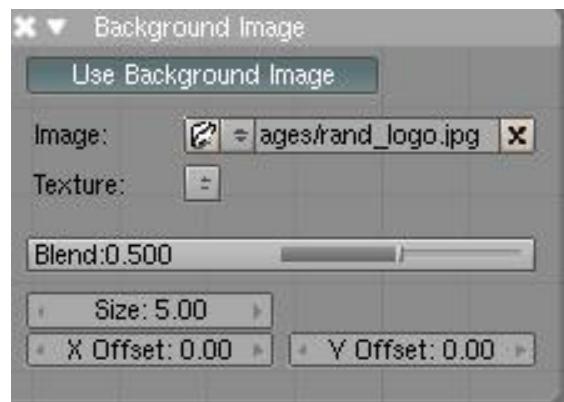
Now the center of your object is at the center of your photo. This is a very important thing because when blender loads in the picture you will need this so that all of your pictures match up with each other 3d. You should repeat these steps with all 3 photos. I also don't recommend doing it in GIMP's “layered mode” as that caused more pandemonium for me. I recommend opening each photo in a new window .

Taking your pictures is the most important part, because if the pictures are not all in the same scale (object size to photo size) then your photos will not line up and you won't be able to place a dot on the same location from front view, side view, and top view.

As a recommendation I would recommend making your first model from a Lego man. That is what I did and it is very simplistic easy practice. To take my photos I took about 10 minutes to construct a photo platform for my object. It consisted of a cardboard box with two sides cut out. I covered the inside area with computer paper. I then used a 2"x4" and a ruler to make sure that the box stayed the same distance from the camera for all shots, as well as marking where the Lego man's feet were positioned inside the box with a pencil. This will provide good pictures, providing you keep the camera at the same distance and zoom for all three photos.



View -> Background Image



2.57 Using Bézier Curve to Model a 3D logo from a 2D logo

2.57.1 Overview

We will be using this graphic as a template for a 3d logo, tracing it, then discarding the 2d image. (*Click thru to get the larger sized version, for you to follow along with*)

2.57.2 Load the background image into blender

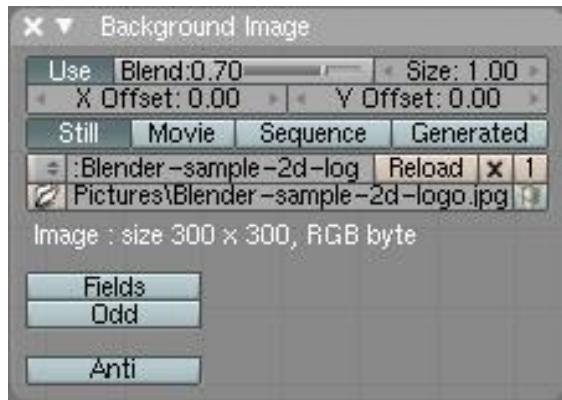
Basics of the background image panel

If you haven't already done so, open blender and select one of the orthogonal view angles by pressing **NUM7** (top), **NUM3** (side), or **NUM1**. (front). At the bottom of the 3D viewport on the left, there are some menus, click

A small window will appear containing just one button marked **use background image**; click this button. A few more buttons will appear. One of them says **image:** and has a small button with a picture of a folder on it; click this button. You are now presented with a file selection screen. Using the navigation techniques from the previous tutorials, find your 2D jpeg image on your computer, click the file in the list once then click the **Select Image** button at the top right of the screen.

Noob note: The image is only displayed in orthogonal view. If perspective view is enabled, toggle to orthogonal view by pressing **NUM5**. (*The image will not be rendered as it is not part of your scene.*)

Once a background is selected you'll have a dialog like this one. (Note: This tutorial was originally generated from Blender v2.37. v2.43 has been added - older versions may differ.) The background dialog buttons are described below:



Details of the background image panel

The **Use Background Image** is a toggle button that turns display of the image on or off. Turning the button off will not clear the settings; it just hides the image. When you turn the button on again, your previous settings are back. Try it - click the button a few times. In v2.43 the equivalent button is the **Use** button.

Image selection is controlled on the row labeled **Image:** **ages\rand_logo.jpg** . There are 2 buttons, a text box, and a final button. The first button is used for browsing for an image. The 2nd button is for selecting an image from a history list. (This will be empty for the first time. Selecting it now will display the image you currently have selected.) The text box **ages\rand_logo.jpg** allows typing in the file directly. The button removes the current background image. Version 2.43 is the same with the addition of the **Reload** button that refreshes the image or movie, and the button which shows the number of users of the image block.

The third row is called Texture and will not be used for this tutorial.

The fourth line, labeled **blend** controls the transparency of the background image with a slider. A setting of 0 is completely solid and 1 is completely transparent. You can adjust it by clicking left or right of the knob for gradual changes, clicking and dragging on the slider for rough settings or clicking directly on the **blend** text for numeric entry.

The use of the blend function will become obvious once we start tracing our logo. For now, play around with it, see how it changes the image, and put it back to the 0.500 default.

The fifth line, **size**, controls the size of the image. This size setting is independent of the zoom for the 3D view window. To see how the size works move the default cube off to the side so that you can see both the cube, the background dialog and the background image. Now watch both the cube and image as you change the size.

Notice how the image changes size but the cube doesn't? Now press **NUM+** and **NUM-** to change the view's zoom. Now both the cube and image change size.

The final row controls the X and Y offset for the image. These controls move the image up and down (Y) or left and right (X). These settings can be useful if you need to reposition the image from the default position. Like the size, these offset values are independent of the view. As you change the offset values the cube you added earlier won't move. Now scroll the view using by clicking and dragging the **SHIFT MMB** and notice how the cube and image move together?

Once you start tracing the image you won't be using the size or offset setting.

Details for version 2.6X

the check box left to the title indicate if you can see the image or not. the *Axis* menu indicate in what views you could see the image.

the button *Image* and *Movie Clip* indicate of course if you use a Image or a Movie.

the button to left of the text box allows to choose the image from the exist, the text box say the name of the image, the F button make the image a fake user, the folder sign allow to upload a image, and the X sign delete the link to the picture.

The **blend** slider mentioned above is now called **Opacity**. The image will be completely transparent when the opacity value is zero, and fully opaque when the value is one.

2.57.3 Delete the cube

Delete the cube (select it, press **XKEY** and select **All** from the **Erase** menu), and set the size so that the entire image is viewable. Then set both the X and Y offsets to 0.

Set the view to top view (**NUM7**) for the rest of the tutorial!

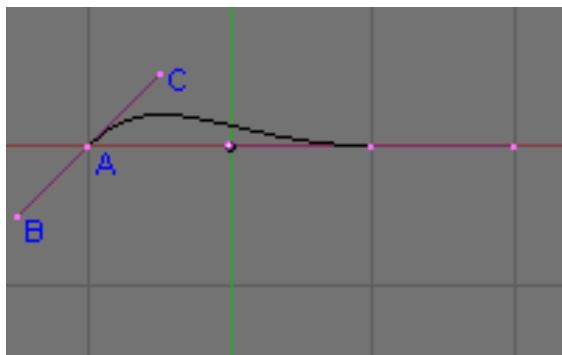
Finally minimize the Background Image dialog. You'll only need it to adjust the blend setting until you finish tracing.

2.57.4 Introducing the Bezier Curve

The **Bézier Curve** allows drawing graceful, complex curves and only requires a few points. Specifically, it only requires 4 points for a curve: two end points and two control points.

For the moment set the blend to 1 on the Background Image dialog. With the center of the 3D view still selected,

press **SPACE -> Add -> Curve -> Bézier Curve**. Alternatively you can use the **Add** menu at the top of the screen or press **SHIFT -> AKEY** to jump directly to the add menu. Be sure to be in EDIT MODE not OBJECT MODE.



The Curve you added.

Unlike the traditional Bézier Curve each Bézier vertex has 3 points. I've labeled the 3 points for the left vector: **A**, **B** and **C**.

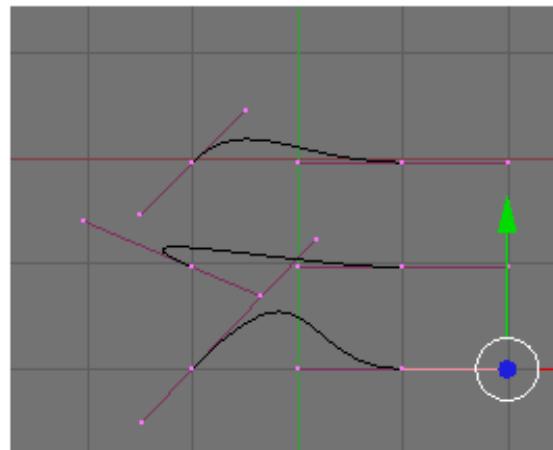
Just like any other vertex you select the control points and the end points with **RMB** and move them with standard commands like **GKEY**, **SKEY**, **EKEY** or **RKEY**. if you wish to rotate, scale, or move the whole curve, then select all the vertices on the curve using one of the various selection methods, and then use the **GKEY**, **SKEY**, or **RKEY**

Point A is an end point. The curve will always go through this point. **Points B and C** are control points. These points influence the path of the curve as it leaves **Point A**. Because the path stops at **A**, **Point B** has no real effect on the path. Instead **B** is currently locked with **C**. (If you move either **B** or **C**, the other will move.) We will fix **Point B** to move independently a little later.

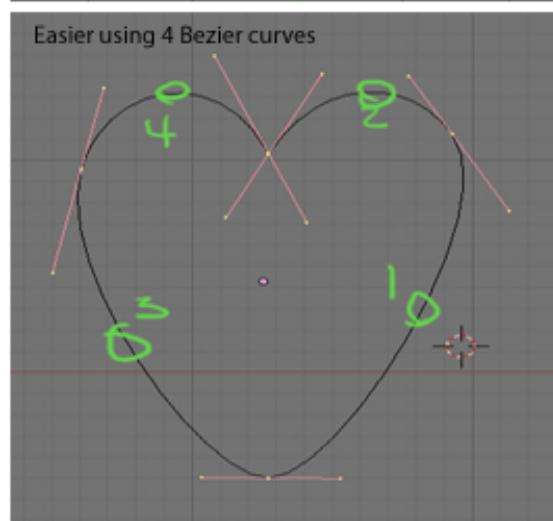
The control points have 2 effects on the path exiting the end point: direction and distance (these are termed slope and magnitude in math circles) from **Point A**. The direction will provide the direction that the path will follow when it leaves **A** and the distance will determine how long the path follows that direction before it starts making its way to the curve's next point.

The example to the right shows how the control points influence the path of the curve. In the top picture, we see three curves. The top curve is the default curve. In the next curve down, **C** has been moved to give a drastically different direction. Notice how the path leaving **A** moves away from the other end point. The third curve, the distance was changed dramatically. Watch the path move much higher than the other two curves.

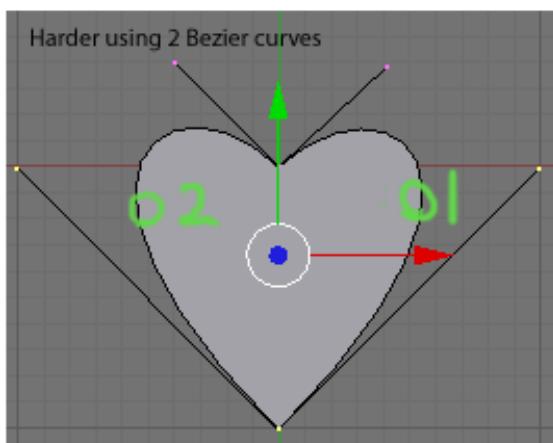
In the bottom example, I've built a heart shape using just the points shown. *Dragging the bottom end point down will make the shape closer to a leaf.* You'll be able to do the same at the end of this tutorial. Go ahead move



Easier using 4 Bezier curves



Harder using 2 Bezier curves



around the points for the curve and see how they all interact. Get a good feel working with the curve and when you're ready we'll move on to tracing.

Now that you know how to work with a bezier curve set blend back to 0.5 on the Background Image dialog so we can start tracing.

2.57.5 Modeling the lighting bolt

Rough Tracing

If you don't already have a curve add one now. It will help to move the curve to the center of the yellow lightning bolt.

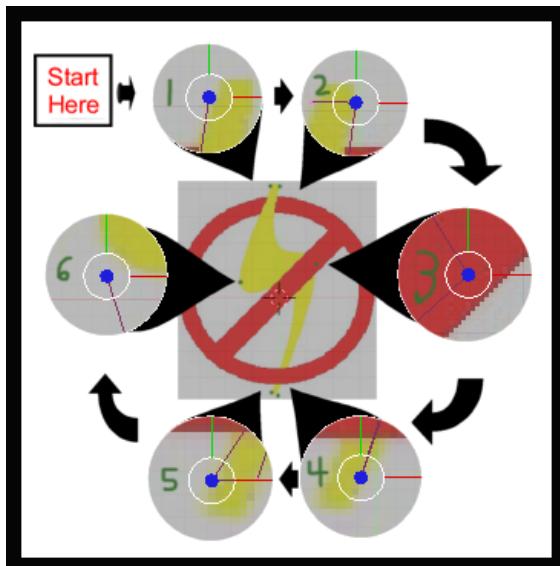


Curve Tools for Blender 3D version 2.37

The first step in tracing is to click the Polygon convert button on the curve tools panel. You'll find this in button on the **Buttons Window**. You may need to select the **Edit Panel**. Press **F9** if this panel isn't visible.

In Blender 2.59, the 3D View panel's Tool Shelf (**T**) houses the Curve Tools, including Set Spline Type, from which you should select Poly. You must be in Edit Mode to reach the Curve Tools.

Noob Note: The 'Curve Tool' panel will not appear if you don't have a bezier curve already placed. I learned this the hard way.

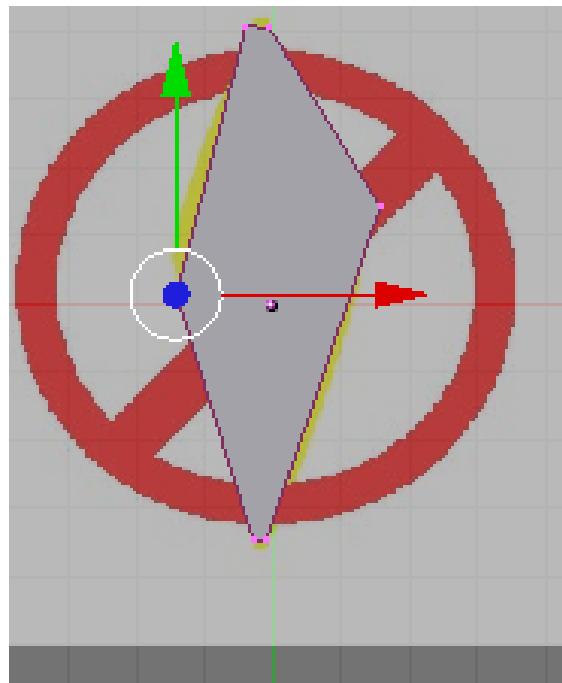


Next, move the vertices of the curve to the points shown in the image to the left. This is called Rough Tracing because you don't need to exactly trace the image.

You only need to approximate the image. Moving the vertices should be done using the instructions from the [Creating a Simple Hat tutorial](#).

Note: Selecting the best place to put a vertex is a bit of an art that you'll acquire as you work with curves. For now follow the arrows along the cutouts and place each of the vertices as shown.

This tracing uses all the vertices of the polygon. Other cases, you'll need add or remove extra vertices by selecting the end point of your curve press **CTRL** and click **LMB**. At the place you clicked a new vertex will appear connected to your curve.



After moving the last vertex, we finish the rough tracing by pressing the **Alt+CKEY** to close the polygon. You should see an image similar to the one on the right. (If you only have an outline switch your view port shading to solid by pressing the **ZKEY** for now.)

Question: how do you close the polygon in blender 2.61?
Answer_ ALT + CKEY pressing c key alone brings up circle select.

Note: In Blender 2.63 and certain prior versions, the curve needs to be changed from 3D to 2D shape in order for the polygon to show up, else you get an error. This can be changed by going into Object Data and, under the Shape heading, switching the curve from 3D to 2D.

Notice how the polygon doesn't cover all of the yellow of the bolt and how in some places the polygon fails to conform to the shape of the bolt. This is expected and should not be a cause for concern. We correct this in the

next section.

Once you've finished several logos you should begin to get a feel for the required placement of vertices. Until then, here are some general guidelines to keep in mind:

- A gradual curve may only require a single vertex.
- Tight curves will likely require two closely placed vertices.
- Curves may not require a vertex at all - you can define some curves using the control points of the adjacent vertices. We did this for both of the inside curves of the bolt above.
- Corners require a single vertex placed where the curve bends. A square, for instance, requires four vertices - one at each corner - to be modeled properly.
- The end point of a curve will always be on the curve. So should all of the vertices you place.

We are now ready to move onto the next step modeling the logo.

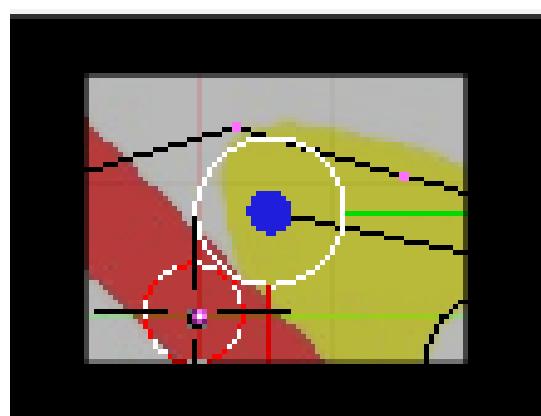
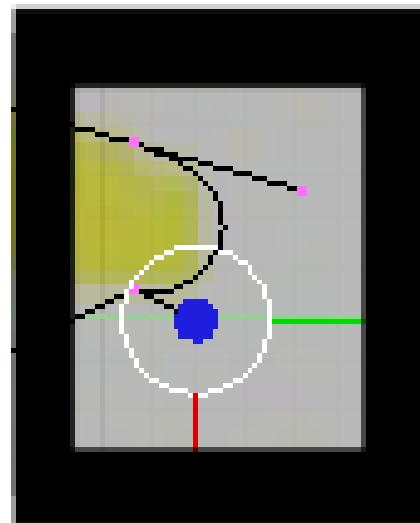
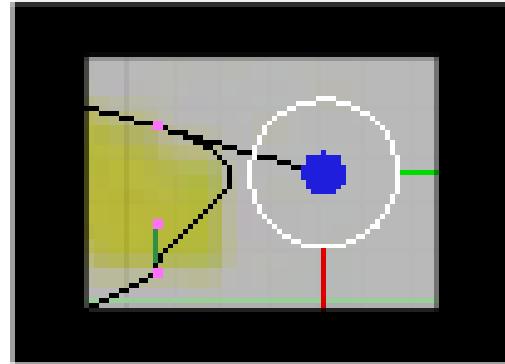
Polishing the Tracing

Press **ZKEY** to return to wire frame mode, then Press the **Bézier** convert button to convert the polygon back to a curve.

This will convert your polygon back into a curve. Nothing obvious will happen. If you look close, you should notice the number of points on the curve tripled. When you converted the curve back to a Bézier curve, Blender changed all of the polygon vertices to Bézier vertices. While the polygon vertex is a single point the Bézier vertex is made of an end point and 2 control points. So the extra points are the control points of the Bézier vertices. These control points are placed along the curve to produce the same shape as the converted polygon.

Our job is to move the control and end points so that the curve follows the edge of the bolt. The trick is to move the 2 control points between adjacent end points to bend the curve to the edge of bolt. First, move the right control point of the top-left vertex. This should pull the curve from its end point to more closely match the line of our bolt. After placing this point, we move to the next control point following a clockwise direction around the bolt. Use the **RMB** to select the point you want to change and move it with **GKEY** to place it.

Noob Note: if you wind up with pink lines when you select the contral handels, that means you hit **HKEY** by accident, and have put the control point in manual. to correct this hit **HKEY** again to put it back into "easy" or auto mode.

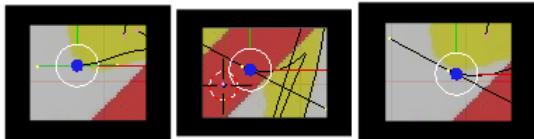


As you move the second point notice how the curve exiting the first end point is drawn away from the edge of the bolt being traced. We now have to adjust the first control point again to get that line back on track. This quickly turns into a balancing act adjusting each set of control points. The trick is to make smaller movements for each iteration of adjustments. Make a game of it and move the control points all along the bolt. Always move along the clockwise direction. This practice is not just for consistency, it keeps your place and ensures that moving a control point doesn't change a portion of the curve that

you've already completed. In time you learn to move the first point only part of the way. Then moving the second brings the curve for the first into correct alignment.

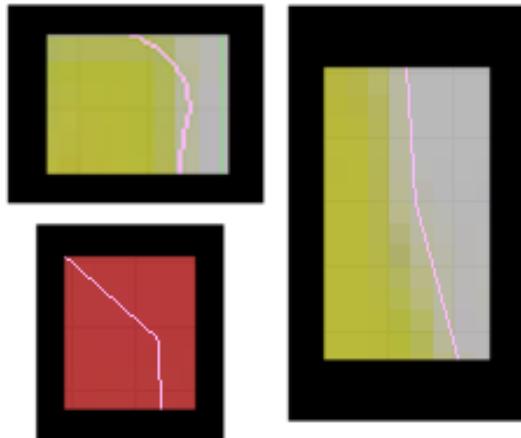
If you have some trouble aligning the curve to the edge of the bolt, consider adding a new point. There are two (at least) ways to accomplish this:

- Select 2 points that surround the problem spot where you want a new vertex and click the Subdivide button on the Curve Tools 1 tool panel.
- If near an end point, Select it, press the **CKEY** to open the curve, then **Control+LMB** click to add a new point beyond the end of the selected final vertex. Press the **CKEY** to reclose the curve.



Adding a new end point

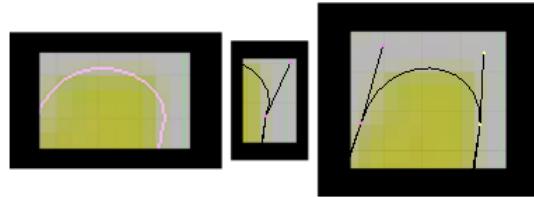
The new end point should be positioned and then you have to adjust the curve on both sides of the end point you move. Any time you move an end point be sure that the curve going into both adjacent (clockwise and counter-clockwise) end points still aligns with the edge of the bolt.



Finding trace defects

Once you've made the complete circuit around the bolt, you're ready for the final polishing of the edge of the curve. Press the **TAB** to switch to the object mode. This makes the polishing easier as Blender hides the points and lines for editing the curve. Now zoom in on the bolt's edge using the **NUM+** or **Control+LMB** drag. Use **Shift+MMB** drag the screen so that you closely observe the entire edge of the bolt while zoomed in closely. Look for places where the curve pulls away from the edge.

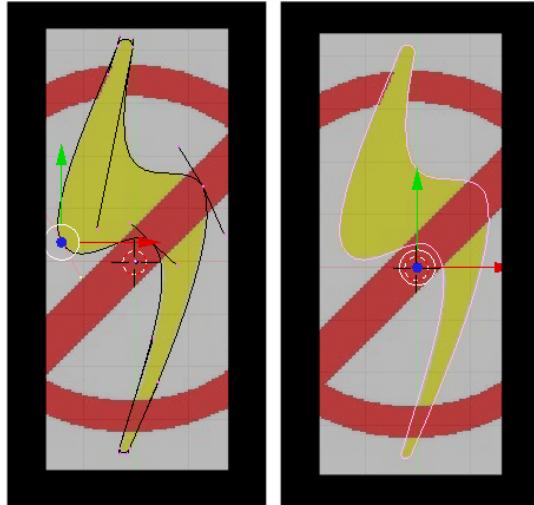
Also look for sharp bends at each of the end points that should be smooth. You can see several defects that I found in my project after tracing the bolt. Switch back into edit mode to fix the curve and then go back into object mode to look for more defects.



Correcting poorly shaped curve

Sharp end points are adjusted by decreasing the angle between end point and the control points. Many times that is impossible to do, without messing up what you're trying to draw, so the other way is to add more curves by subdividing (**WKEY** -> subdivide) the existing curve between two end points, and playing with the control points to get a nice smooth curve,

Places where the curve pulls away from the edge can be resolved by moving the control point closer to the edge. In the above image the curve was found to have been pulled away from the edge. This was fixed by moving the control point a little to the left.



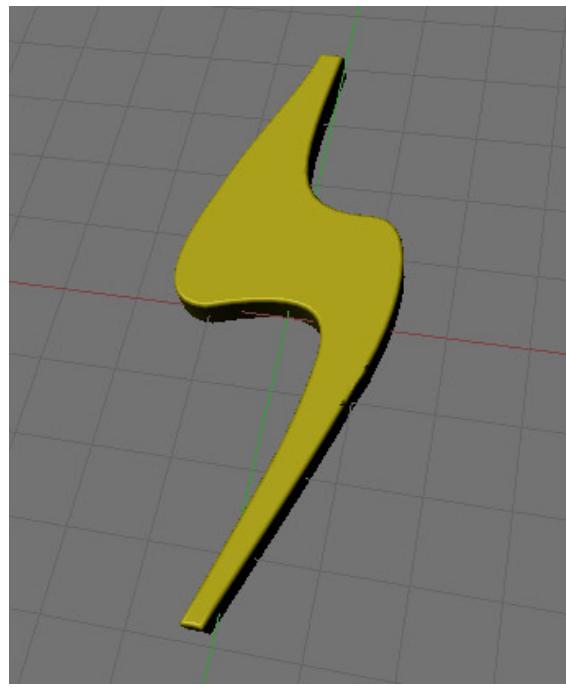
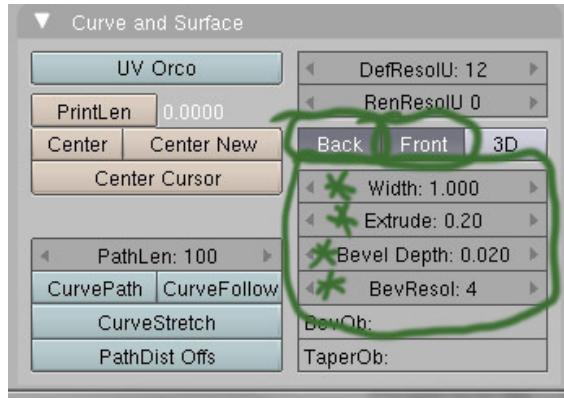
Final Polished Curve

Here's the final polished curve for my project. It is shown in both edit mode and object mode so you can clearly see both the control and end points on the left and the curve to the right.

Noob note: In blender 2.37 and later (not sure of earlier versions) pressing the **HKEY** toggles the control points between free and aligned (Edit Mode). Free Control Points are good for sharp angles, and aligned are good for smooth curves. This shortcut is in the **Space -> Edit -> Control Points** menu.

This concludes the tracing of the bolt. All that remains is making the curve 3 dimensional, applying a material and positioning the final object. Before doing that, we will trace the circle in the next part of the tutorial. Save this project if you want to take a break before continuing. You'll need it on the next tutorial.

Adding a Third Dimension



First, give the object some depth. If you are in wireframe, hit **ZKEY** to go into solid mode, then down on the edit-buttons screen, locate the “Curve and Surface” panel, and set the following values:

- Back and front buttons are selected (depressed), otherwise it will be see through.
- Extrude: 0.2 (the height of the extrusion on either side)
- Bevel Depth: 0.02 (the radius of the round bevel applied to the extruded edge)
- BevResol: 4 (the number of subdivisions on the bevel curve)

Noob note: In previous versions, Extrude and Bevel Depth were Ext1 and Ext2.)

Noob note: In versions 2.6x (possibly also 2.5x), go to *Properties panel --> Object Data --> Geometry* for these settings.

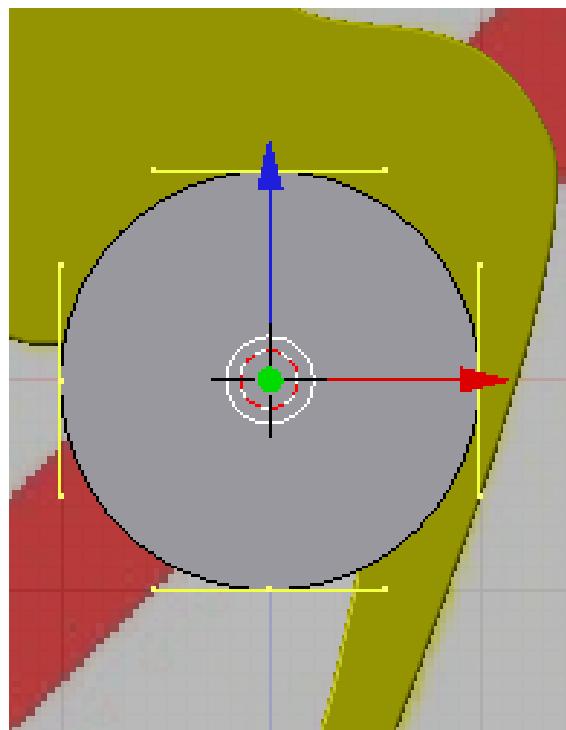
Now you can use your knowledge from earlier in this book to change the material and/or add texture to your logo. Feel free to rotate, add lighting, or whatever floats your boat. Don't forget to press **ZKEY** to toggle wireframe mode.

You should now have something that looks like the photo on the left.

to it before continuing.

Method one is the easiest, so try it that way first.

Method 1 - using bezier circle



Bézier Circle

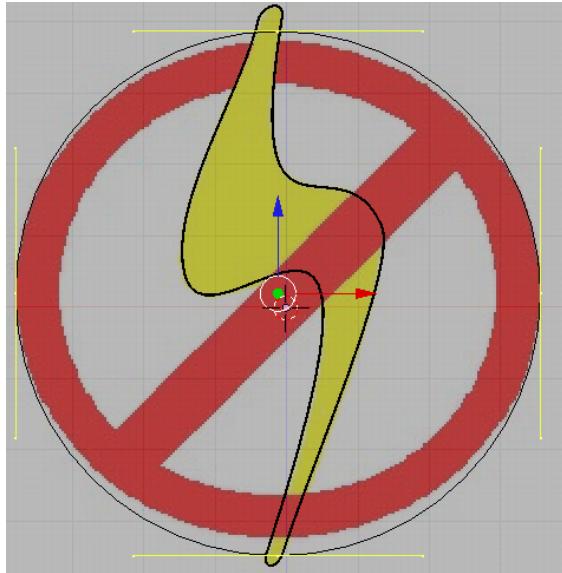
2.57.6 Modeling the red circle

It will be helpful to make the lightning bolt distinct from the circle part of the logo, by applying a yellow material

Outer circle Switch to object mode by hitting **TAB** if you aren't already there. **LMB** somewhere near the

center of the background photo. Press **space -> Add -> Curve -> Bézier Circle** to add a closed bezier curve with four points forming a circle. If you are in solid draw type, switch to wireframe with **ZKEY** so you can see the underlying image better. Hit **SKEY** to scale the bezier circle to fit over the circle in the image. Again, if you are using one of the newer versions, you will have to switch the circle from **3D** to **2D** in the object panel.

Noob note: You should stay in object mode for now, as it's easier to scale and move the bezier circle without having to keep track of which end points are selected.

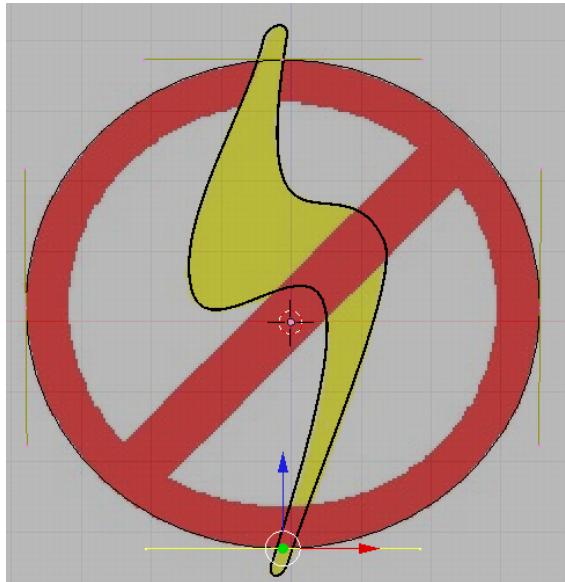


Match up left and right sides

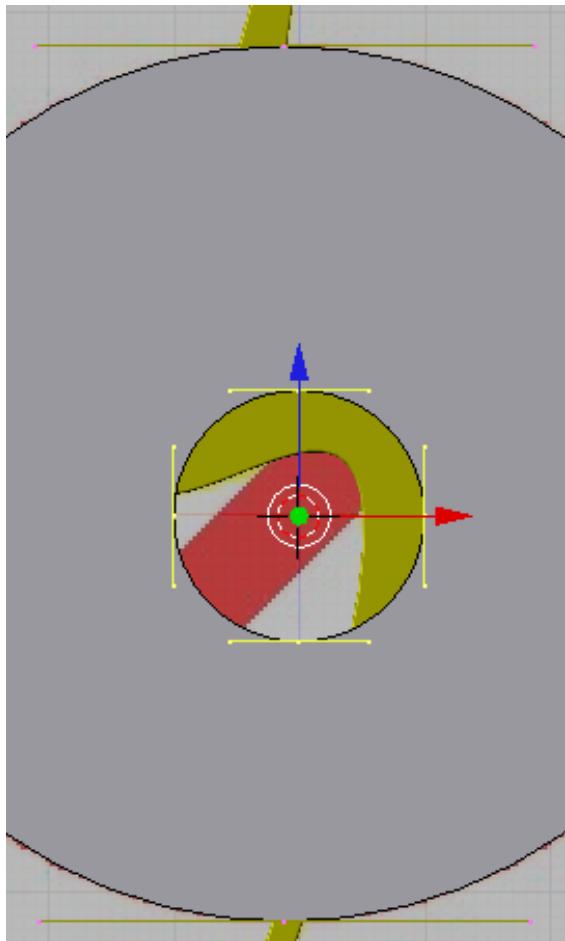
You will probably find that the bezier circle is not dead center on the sample logo so you will need to move it with **GKEY** to center it. You may need to scale it and move it several times to get it right. You will also find that the circle in the sample image is actually a slight oval, so scale and position the bezier circle, along the X axis, and Y axis, so that it touches the circle in the image on the left and right sides.

Normally, you could then scale the circle and constrain it on the X and Y axis by using **SKEY**, but it turns out that the oval isn't regular anyway, so just select the point on the top and hit **GKEY** and then **XKEY**, or **YKEY** to move it around until it touches the top of the oval in the image. Then do the same for the bottom point and you should have a pretty good fit.

Inner half circle 1 Just to understand what's happening in the next steps, switch to solid view with **ZKEY**. As you can see, you now have a circle, but it's filled in the middle where you want to be able to see through it. To cut a hole out of the circle, be sure that the circle is selected, then hit **Shift+S -> cursor to selection, to put the cursor in the middle, switch into edit mode**, if you're not already in it, and then **Space -> Add -> Bézier Circle**. A new



Move top and bottom points

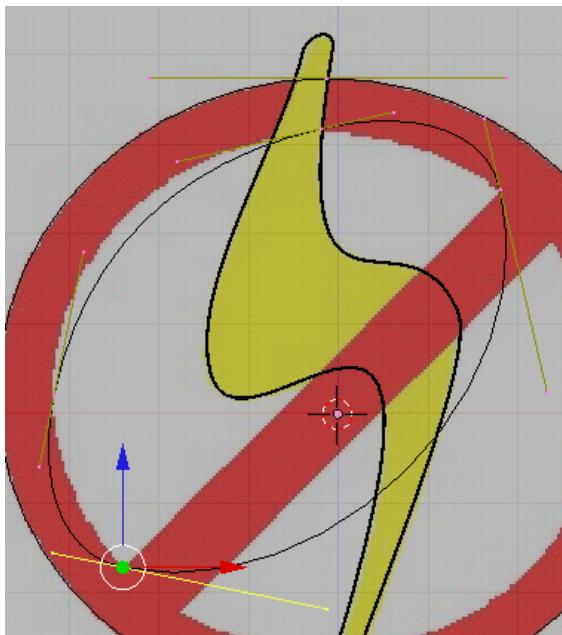


Circle in circle

circle will appear inside the larger circle. As you can see in solid mode, the new circle actually cuts its shape out of the larger circle surrounding it.

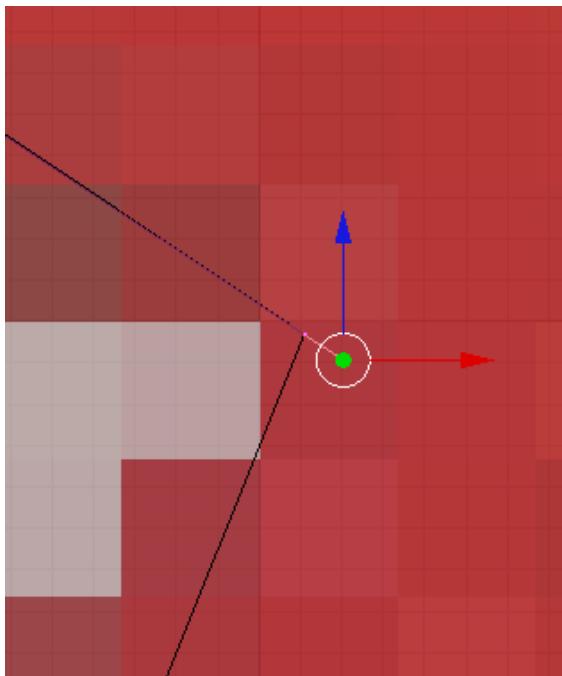
Noob note: make sure you are in Edit Mode or it will add

the new Bézier circle without cutting it out of the existing one.



Place two points at either end of the bar

Switch to wireframe mode with **ZKEY** so that you can see the underlying image again. Scale up the smaller circle so that it approximately fits the inner part of the circle in the image. Don't worry about getting it exact since you'll be manually moving all four points anyway. Move the bottom point of the bezier circle to the top left corner of the bar that crosses the circle.

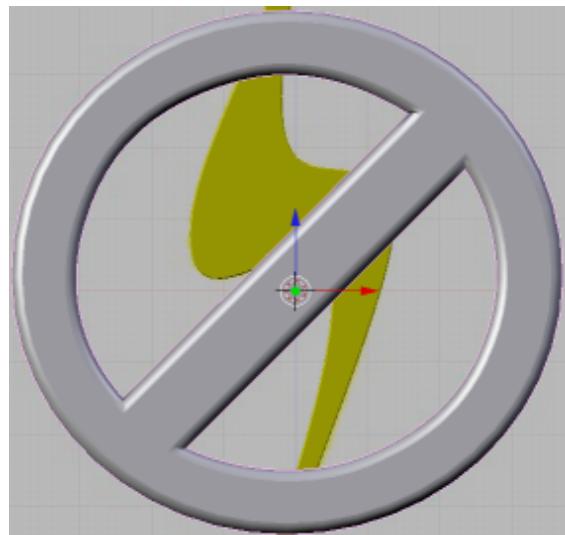


Place two points at either end of the bar

Move the right point of the circle to the other corner.

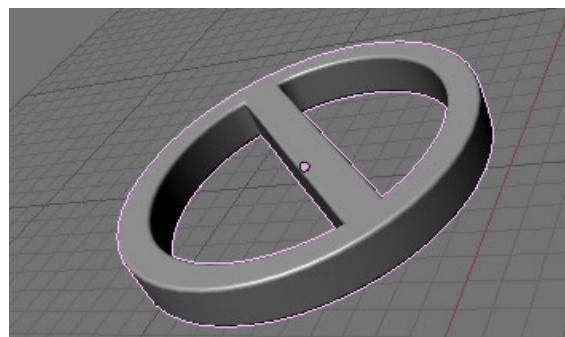
When you create a Bézier Circle, Blender by default sets the alignment of all the control points to aligned. To make the diagonal bottom edge you need to break the alignment on the two lower sets of control points. Hitting the **HKEY** will toggle between free control points and aligned. Once you've selected the two lower bezier points and hit the **HKEY** to make them free you can move each of the inner control points to create a nice straight edge. Then move the other two points and adjust their control points until you have a pretty good approximation of the rest of the inside curve.

In Blender 2.63 and others, pressing the H key seems to hide vertices instead. To set control points to free, select the vertex, then in the curve tools under Handles, select Free. (can also be set via the menu at Curve-control points-set handle type-Free)



Wider bevel has a nice look

Inner half circle 2 Next step is to press **space -> Add -> Bézier Circle** again and repeat the same steps, but for the lower opening in the logo. Once you've completed both openings, switch back to solid view with **ZKEY** and examine your work. Make any adjustments you need to by switching the draw type back and forth as needed.



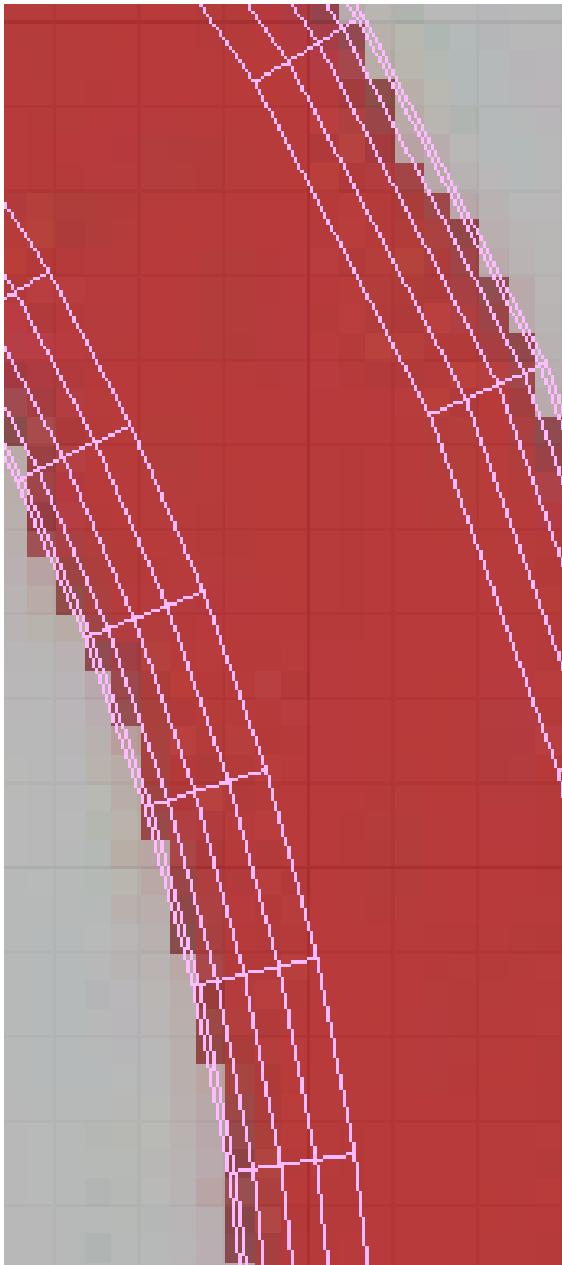
Make it 3d The next step is to make this part three dimensional like you did with the lightning bolt. Go to object mode with **TAB**, then select the editing buttons.

Under curve and surface, set:

Extrude/Ext1 to 0.05,

Bevel Depth/Ext2 to 0.02

BevResol to 4.



Width adjustment restores to normal size

Now that we have beveled it there will be a problem. Switch back to wireframe mode with **ZKEY** and you'll see that the bevel has widened everything so that the circle no longer matches the original image. This can be fixed fairly easily by reducing the width parameter under curve and surface to about 0.98.



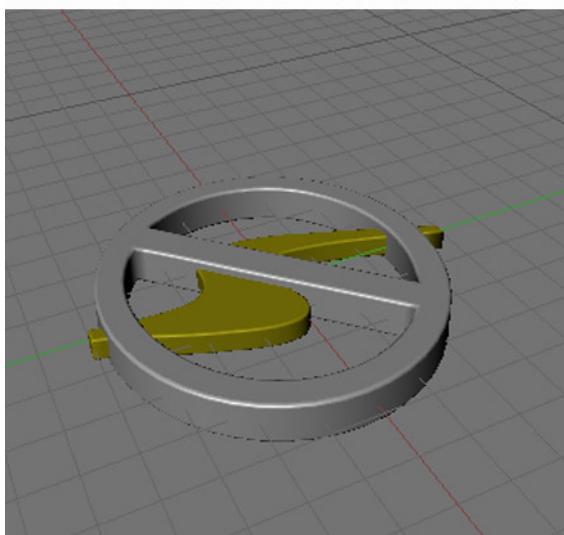
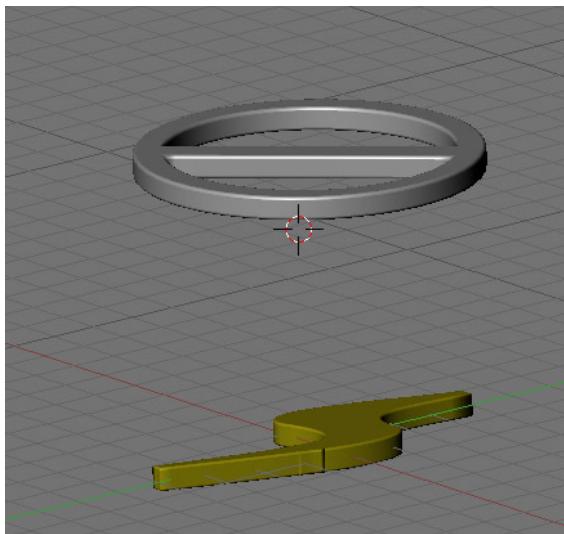
Bevel has increased logo size

Noob Note: I don't see an easy way to adjust the width parameter under curve and surface in Blender 2.70. I was, however, able to adjust using **WKEY**, Width, and then incrementing with the mouse.

For final steps, select the lightning bolt again and switch into sideview with **NUM3** and hit **GKEY** and then **ZKEY** to move both elements up or down until the bolt is inside the circle.

Apply a red material to the circle and bar portion. Finally, you can go to view, then to background image and hit the background image button to hide the image now that it is no longer needed. At this point, you can add any finishing touches for lighting and camera angles and render the logo.

After rendering you will have something like this:

*Final render*

Method 2 - using mesh circle and edges

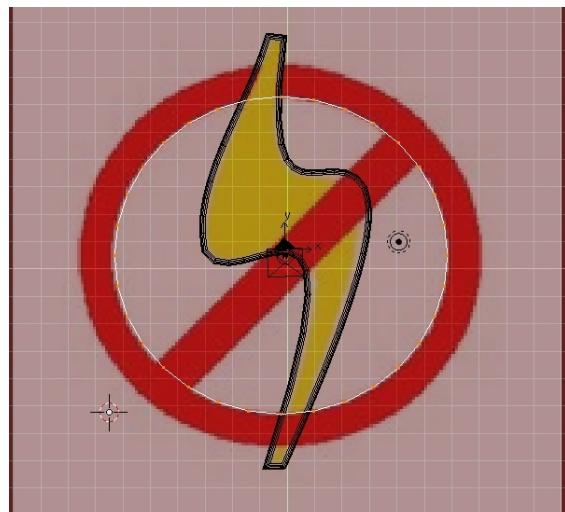
Instead of using the Bézier Curve, you can use Circles.

There are two ways of doing this:

Easy way Place two vertices (CTRL+LMB), one on the inner circle and one on the outer circle so that they form a line. This line has to be perpendicular to the circle you are tracing. Then place the cursor on the middle of the circle. Then just use the spin tool under the Mesh tools tab (360 degrees and 32 steps). This creates a circle made of 32 adjacent squares. Give it more steps to increase the quality of the circle.

Detailed way Go to the top view (NUM7), and press SPACE -> Mesh -> Circle. Accept the 32 vertices, you can make it less but it won't look as good.

Move it into the center of the circle, if you don't then I advise you have wireframe on for the moment (press Z). Then press S, for scale, and make it the correct size for the inside of the circle. Once you have that in the correct place, like so:

*Adding the first circle*

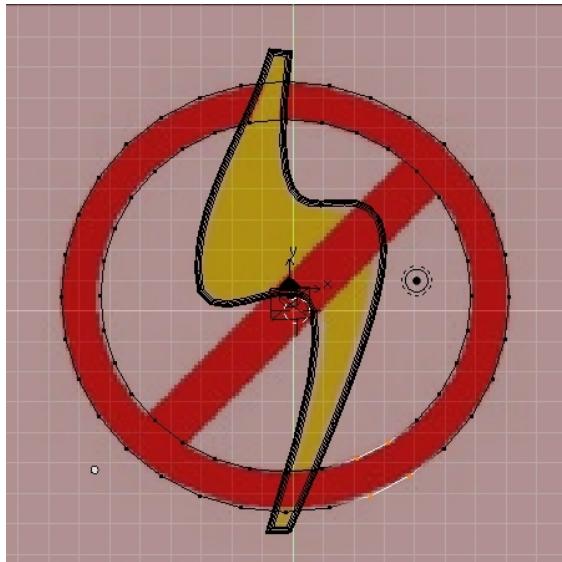
You may need to stretch it sideways a little, then press 'E' to extrude (choose Only Edges), press 'S' to scale and another sized circle will appear, size this appropriately then click LMB.

Deselect the second circle, then select four vertices that are near each other and that form a square.

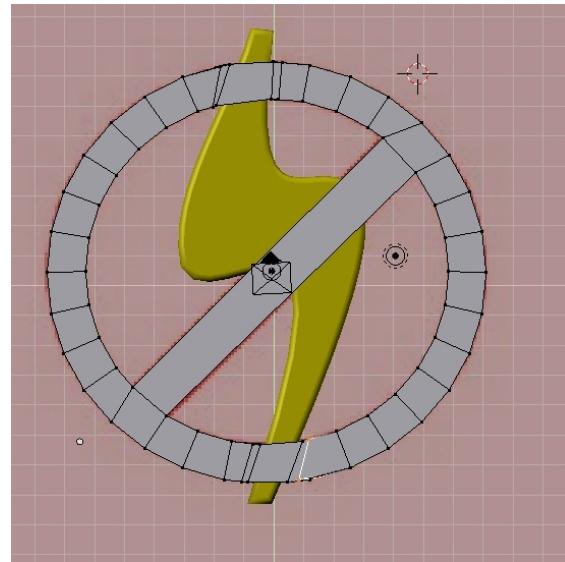
Now press F and a face will appear, i.e. the box will be filled (turn off Wireframe, press z). Now do this right around the circle. To do this, hold down Shift and Ctrl, then draw a circle around the two vertices you wish to deselect with the Left Mouse Button. Then draw a circle around the next two vertices while holding down Ctrl and LMB, **NOT SHIFT**. Shift changes the control from selecting, to deselecting.

Once you have gone all around the circle it's time to make the line through the middle. To make the crossing line you need to move 4 of the vertices slightly; example below:

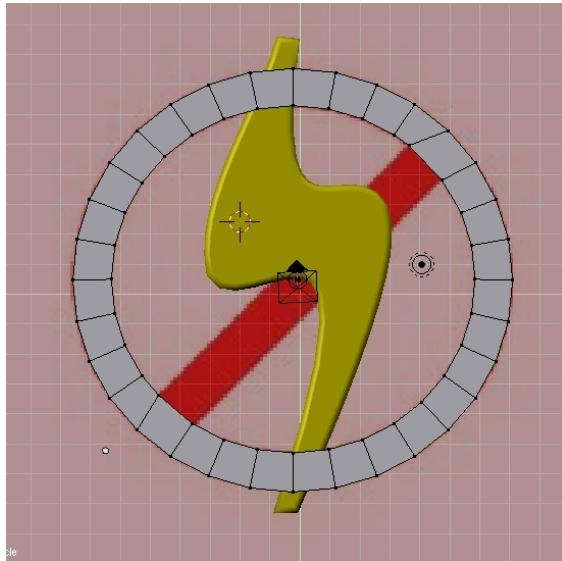
Once you do this, highlight the 4 you moved, then press



Selecting the four vertices

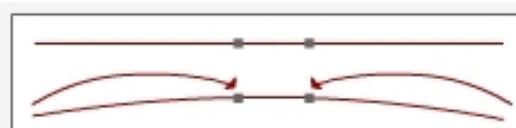


Creating the overlap



Moving the four vertices

circle. Change to Wireframe (z) if you are not already in it then highlight the 16 vertices of these boxes and raise them. Now do the same for where the center line crosses the bolt but create 4 lines instead of two. To explain why, I've created a diagram.



With two points the line bends from the end, to the dot.



Now the line bends from the other two points, this is the effect we want.

F.

Why we use 4 points instead of 2

Making the circle 3D Highlight the full circle by pressing A either once or twice. Go to Side View, and press E for extrude and drag it down so that it is the same thickness as the lightning bolt (you'll see why).

Now look at what you have made... it looks nice enough but where the lightning bolt goes through the circle it looks a bit odd so we will make it look like the circle is laying on top of the bolt. Where the bolt goes through the circle, note the edges and the vertices. Move them so that the lines are just either side of where the bolt goes through. Then make new edges using CTRL+R on the outsides of these edges. Like so:

Now you have squares where the lightning bolt hits the

Now we can make the finishing touches, add subsurf and set it to 2 or 3 and add color! Then you are done :-) I won't go over subsurf and adding color because people have covered that better than I could in previous 'Noob to Pro' pages. I think that's everything.

You should now be looking at something similar to:

Method 3 - using mesh circle and a cube

Tracing the **No symbol** is somewhat more complex than the lightning bolt. The reason is this symbol is hollow and requires additional planning to trace than just following an outline of an object.

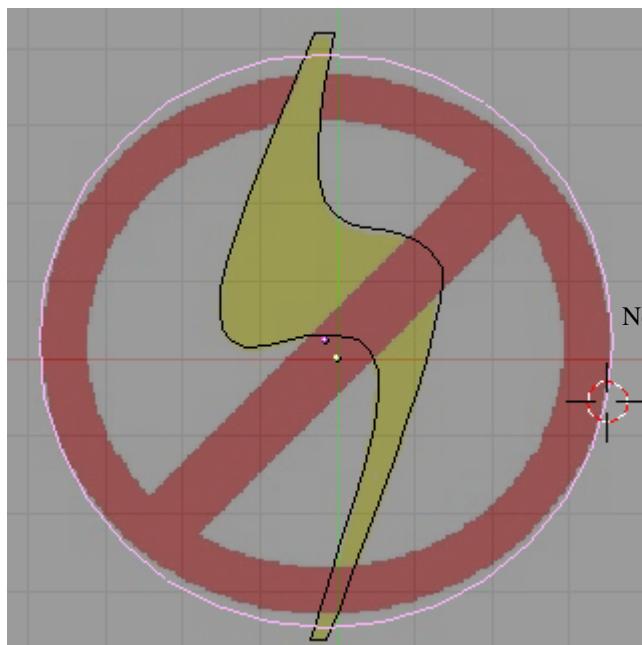


The finished article

I'll be back to explain the difficulties in the near future, so stay tuned.

One way to do this is outlined below:

- The first thing to do is add a mesh circle. Hit **SPACE** and select 'Add -> Mesh -> Circle'. (This circle is mostly for measuring purposes, and will ultimately be removed.)
- Scale (**SKEY**) and move (**GKEY**) the circle so that its sides are even with the sides of the logo circle, and the top and bottom are an even distance from the top and bottom of the logo circle. (The logo circle is not perfectly round.)



- Make sure you're in object mode (**TAB**), add a cube (**SPACE**, 'Add -> Mesh -> Cube') and place it (**GKEY**) on the right edge of the circle so that it is centered on the right edge.

- Scale (**SKEY**) the cube so that it is the same width as the wall of the circle and touching each side of the wall.

- Now, we want the 3Dcursor at the center of the logo circle. To do this, go back to object mode (**TAB**), and select the circle that we added earlier (**RMB**), then press **SHIFT+SKEY** and choose 'Cursor -> Selection'. (This is the main reason why we have this circle here.)

- Reselect the cube (**RMB**) and **TAB** to edit mode.

- Choose **NUM7** for the top view, and use **RMB** to select only the front face of the cube. (Before this you have to make sure that you are in 'Face select mode'.)

- Press **XKEY** and choose Erase Vertices. This will reduce the cube to a square

- Press **NUM7** to go to the top view then select the Editing panel (**F9**).

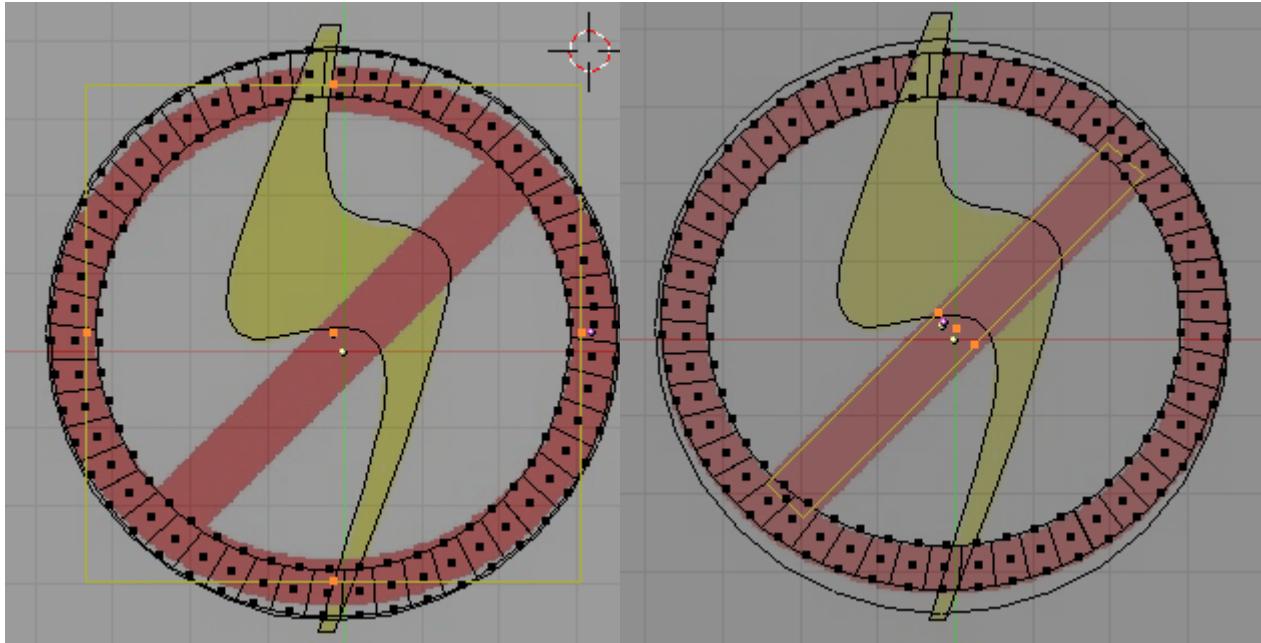
- Press **AKEY** to select all, and then make sure that the rotate settings are set to degr:360 , Steps:50 , Turns: 1. (Actually, Steps can be whatever number suits your fancy. You may want to play with various values.) This is just like we did for creating the man's hat.

- Click on 'Spin' to extrude the square into a ring.

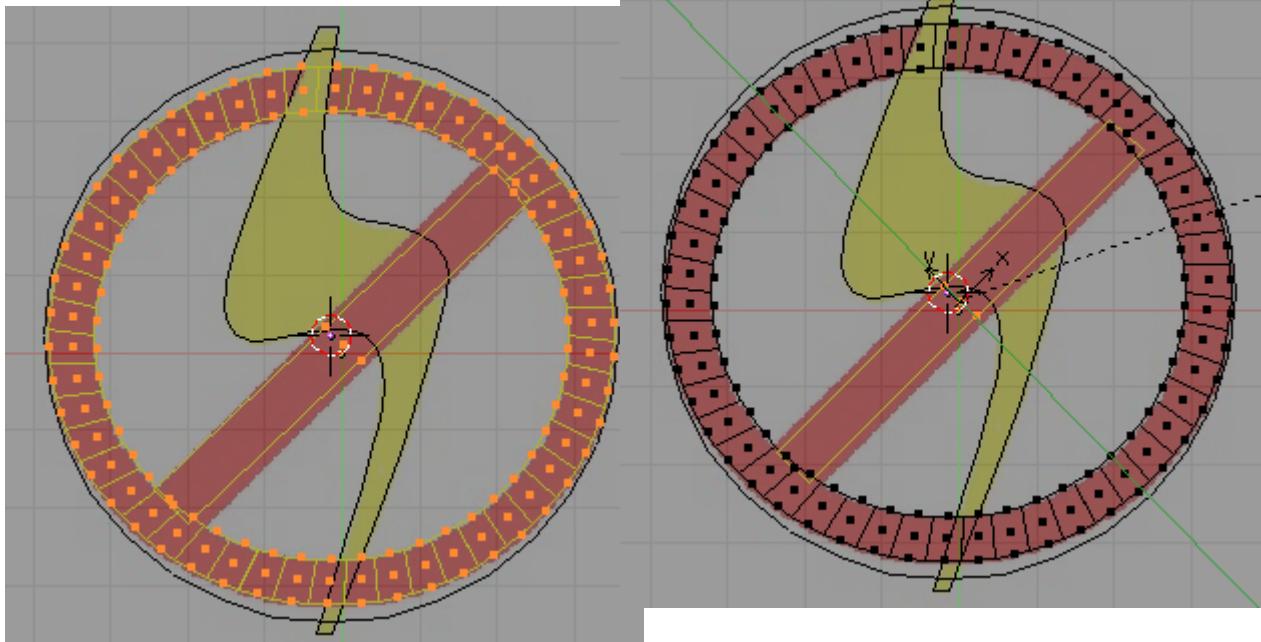
Next we will create the bar.

- Staying in Edit mode, hit **SPACE** and select 'Add -> Cube'. If you haven't moved the cursor since the ring extrusion, it should appear in the middle of the ring.

- Expand (**SKEY**) the cube so that it just encompasses the inside circle of the ring



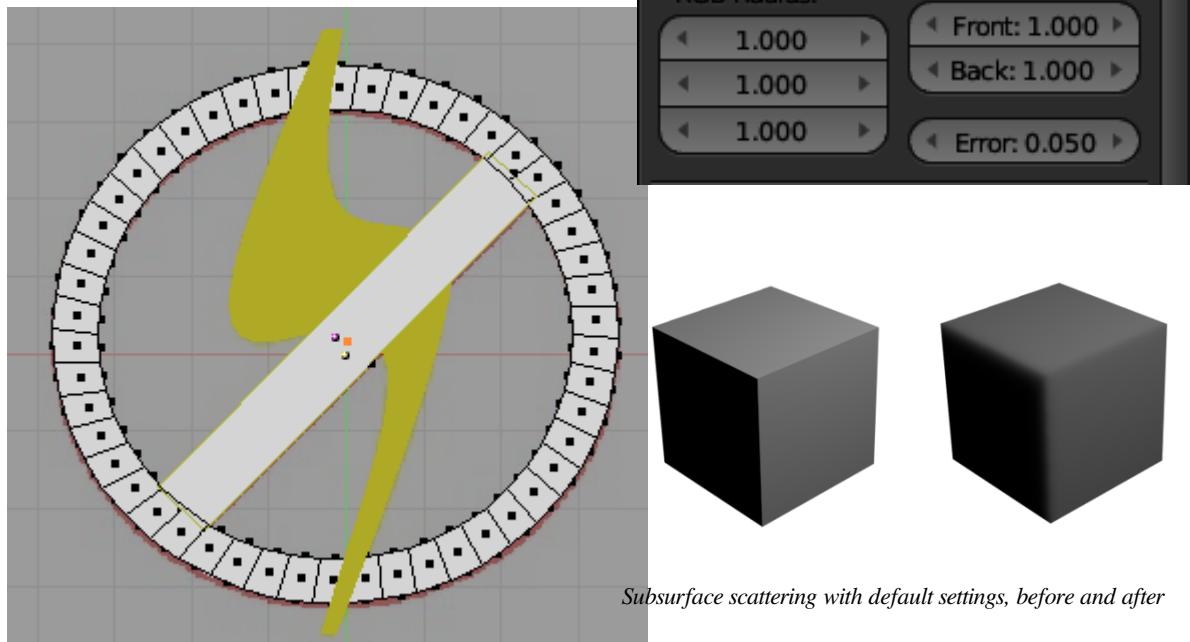
- Scale (**SKEY**) the square in the Y axis (**YKEY**) so that it is approximately the thickness of the logo bar.
- Return to object mode (**TAB**), and rotate (**RKEY**) the ring together with the just designed bar so that it is parallel to the logo bar.
- In Edit Mode (**TAB**), squash the ring in the Y direction (**SKEY**, **YKEY**) so that it fits the logo ring.
- The next thing we need to do is scale the bar so that it is precisely the same width as the logo bar.
- To do this, choose scale (**SKEY**), and press **YKEY** twice. This will go to local scale mode (local to the object). This is why we were rotating the entire ring, since the ring and bar are part of the same object.



- In edit mode, reselect just the bar. To do this, choose 'Face select mode', and do an area select of the face selectors near the center of the bar (use the **BKEY**, or **SHIFT+RMB** methods).
- Re-rotate (**RKEY**) the bar (in Edit Mode) so that it is parallel to, and in the middle of, the logo bar.
- We no longer need the original circle now, (it should be sticking out from the top and bottom of the ring), so select it (**RMB**), and delete it (**XKEY**).
- If you hit **NUM3** You'll see that the bar and the ring are at completely different heights than the lightning bolt. Select and scale them so that they are all roughly the same height (the bar should be slightly

higher than the bolt, and the bolt slightly higher than the ring so that all the proper parts are covered.).

- You can use the trick above to select (RMB) only the bar, scale it in the Z axis (SKEY, ZKEY) and then hit SPACE and choose 'Select -> Inverse' to select the ring (all but the bar).
- When done rescaling, press **NUM7** to look at your handiwork.



Subsurface scattering with default settings, before and after

- There is one last bit, which is to slightly rotate the whole ring around the X axis so that it is below the top of the bolt and above the bottom.
- Once that's done, all that's left to do, is color in the ring.

2.58 Subsurface Scattering

Real-life materials which may look opaque are often not perfectly opaque: light may penetrate a little way into the surface before bouncing off. This is noticeable as a subtle softening and colouring of the edges of shadows on the object. This effect is known as *subsurface scattering* (commonly abbreviated "SSS").

Open a new default Blender document. Select the default cube. Go to the Material  context in the Properties window. Look for the Subsurface Scattering panel. Check the box at the top to enable it for the cube's material.

Compare how the cube looks with and without SSS enabled: see how the edge of the shadow becomes a little bit fuzzy? This is simulating the effect when the light penetrates a little way into the material, emerging just

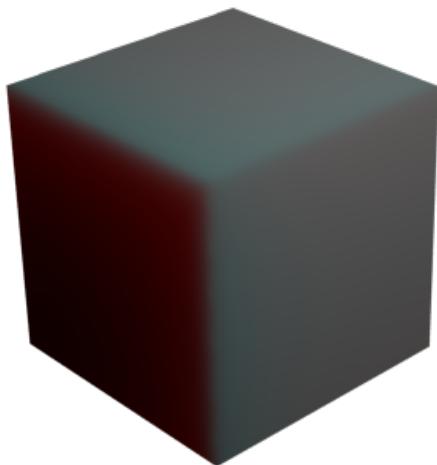
within the edges of the shadow, making them that little bit lighter.

Note the following settings in the SSS panel:

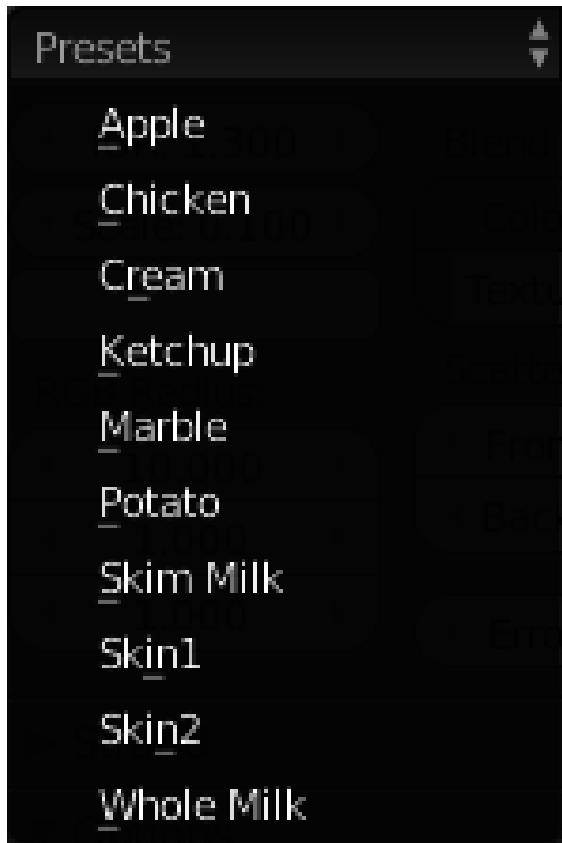
- The *Scale* controls how the overall size of the effect relates to the size of the object. If you size your object so that 1 BU is equivalent to 1 metre, then the default scale of 0.1 should produce a realistic effect.
- The colour swatch causes the scattered light to take on the specified colour.
- The RGB Radius values govern how far the red, green and blue components of the light penetrate into the material before being scattered. These are relative values, all subject to the overall Scale factor.

For example, this is what happens when the red radius is increased to 10, leaving the green and blue radii and the actual scattering colour unchanged: this causes the red light to travel further, tinting the interior of the shadow red and the adjacent area the opposite colour—blue-green. This is similar to what happens with human skin, as the light scatters through the blood vessels underneath.

Try the various options in the Presets menu: how convincing do they look? Of course, they may look better if you apply them to a model that is supposed to look like the actual material.



Higher red scattering radius



2.59 Ray Tracing

You previously learned about using diffuse and specular shaders to control the appearance of a material. These settings only affect how the material reflects *direct* light from lamps; but in the real world, objects are also illuminated by *indirect* light bouncing off other lit objects.



A particularly outstanding ray-tracing example, from the Wikipedia article

In particular, there is the important case of light bouncing off mirrors and other glossy or polished surfaces before travelling to the camera, and also of light passing *through* transparent objects and being *refracted* (bent).

In the real world, such light can illuminate other objects, and it can also do so after bouncing off non-mirrorlike surfaces. The general problem of modelling such indirect lighting is called *global illumination*, commonly abbreviated “GI”. Unfortunately, the Blender Internal renderer, which is what we have been working with so far, cannot deal with GI in its full generality; to cope with that, we would have to use the Cycles renderer, which is introduced [later](#). So for now we will confine ourselves to mirrorlike reflections which do not illuminate other objects.

2.59.1 Setting The Scene

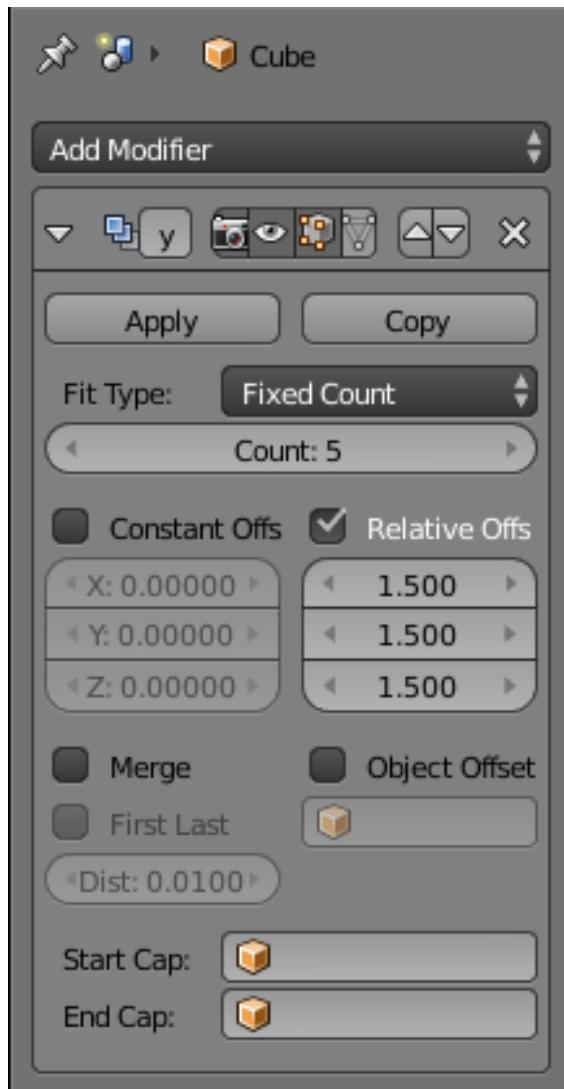
Let's create a very basic model we can use to experiment with various ray-tracing effects. Open a new Blender document. Make sure the default cube is selected. Go

to the Modifiers context in the Properties window, and add an Array modifier to the cube.

The cube will immediately become two cubes, but so close together that they look like a single cuboid. This is because the default settings for the Array modifier are to use a Relative offset of $(X, Y, Z) = (1.0, 0.0, 0.0)$. Change these offsets to $(1.5, 1.5, 1.5)$, so the copies are more widely separated at a more interesting angle, and increase the repetition count to 5.

Now try rotating and repositioning the camera and the cubes to get them all in the view. If you hit F12 to render now, you should see something like at right. So far not very exciting. But since we are going to be dealing with reflections and refractions, it would be nice to have something in the surroundings to be reflected and refracted.

One easy thing to add is a world texture. Go to the Tex-



ture context in the Properties window. Make sure the World texture (the leftmost of the 3 buttons at the top of the topmost panel) is selected. Create a new texture, and change its type to “Magic”, which has a nice variety of different colours. Go to the Influence panel; by default the top-left checkbox (affect background progression) is checked; but this does nothing, because the default sky setting is not to have a progression at all. So uncheck that box, and check the second one on the left

(affect horizon colour) instead.

Also under the “Size:” setting, set the X, Y and Z scaling all to 3.0; by shrinking the pattern, this will bring more detail into the view.

Now if you hit F12 , you should get an image with a slightly more interesting background.

OK, we are ready to start playing with the material settings...

2.59.2 Ray-Traced Transparency

In physics, *refraction* is what happens to light when it crosses the boundary from one material (e.g. air) into another (e.g. water); it slows down when it enters the denser material, and speeds back up when it leaves. The *refractive index* of a material is a measure of how much the speed changes relative to a vacuum (where light travels at full speed). As the speed changes, the light beam also changes direction, giving rise to well-known “bending” effects like you see when you put a teaspoon in a glass of water, or look down into a swimming pool (as at right).

Anyway, back to our tutorial model. Make sure the cubes

are selected. Go to the Materials context in the Properties window. Find the Transparency panel, and check the box at the top to enable transparency. In the row of buttons for selecting the type of transparency just below that checkbox, titled “Mask”, “Z Transparency” and “Raytrace”, select “Raytrace”.

In the editable fields immediately below those transparency-type buttons, the one at the top left is titled “Alpha:”; you will have to reduce it below its default value of 1.0 in order to see any actual transparency effect. Try reducing it to 0.5.

Just a bit further down from the Alpha field, look for two more fields: “IOR.” (“Index Of Refraction”) and “Filter.”

To simulate glass, set the IOR to 1.5 (other useful values are 1.33 for water, 2.4 for diamond etc). The Filter value controls how much of the diffuse colour of the material the light takes on as it passes through; set it to something like 0.5, though you probably won’t notice much effect from this unless you specify a strong colour in the diffuse shader settings.

The “Depth:” setting controls how many times the light passes across material boundaries before the renderer gives up keeping track. Larger values give more realistic results, at the usual cost of increased render times. In more complex scenes where you have transparent objects in front of other transparent objects, this value will have a definite effect; here it probably doesn’t matter too much.

Now if you hit F12 , you should see something like at right. The cubes still look a bit grey; if you set the alpha to 0, they will look much more transparent.

Also try playing with the “Amount:” slider under “Gloss:”: the default value of 1.0 gives perfectly smooth refraction, while values less than 1.0 give a “frosted glass” effect, blurring the light as it passes through the material.

2.59.3 The Fresnel Factor

Real-life materials are never perfectly opaque or perfectly absorbent; even with something like a shiny metal which seems entirely opaque, light still manages to penetrate a little way into the surface, and even with the blackest of black soot, there is still some (tiny) amount of reflection going on.

The general behaviour is that a material is most transparent when its surface is viewed directly-on, and it is most reflective when it is viewed almost parallel to the surface (as usual, Wikipedia has all the [gory details](#) if you’re interested).

If you look back at that Transparency settings panel, you will see to the right of the Alpha field one labelled “Fresnel:”, and another one below that labelled “Blend:”, which comes to life when the Fresnel value is set to something greater than its default of 0.

The Fresnel (pronounced “fray-nel”) value is the *power*, which governs how sharply the transparency of the surface changes with viewing angle; 0 means the transparency stays unchanged at all angles, while higher values cause the transparency to fall off more rapidly towards the edges. The maximum value you can set for this field is 5.0.

2.59.4 Ray-Traced Mirroring

Just below the Transparency panel in the material settings, you will see the Mirror panel. Check the box at the top to enable it. Note the “Reflectivity:” editable field just below the checkbox; you will need to set this to something other than its default of 0.0 in order to actually observe any mirror effect.

If you render now, you should see something like this.

See also the colour swatch specifying the colour of reflections: the default white means that reflections keep their colours unchanged, which is characteristic of reflections off glass, plastic or ceramics. Reflections off metal tend to take on the colour of the metal.

The Gloss setting governs how mirrorlike the reflections are: reducing this from its default of 1.0 adds blurring to the reflections, giving the effect of less-polished surfaces.

Note the Mirror panel has its own Fresnel settings. Go back to the Transparency panel, and set the Alpha to 0. That will disable the diffuse and specular shaders completely, giving us a pure ray-traced material, and also render the transparency Fresnel ineffective. Instead, we will control the Fresnel here in the Mirror panel. The be-

haviour for a nonzero Fresnel power is similar to before; however, instead of fading from transparent in surfaces viewed face-on to effects from the shaders when viewed edge-on, it will fade from transparent to mirrored.

Set the Fresnel value to something like 2.5. Compare the render to the previous one without Fresnel:

Finally, set the Reflectivity to 1.0, and the Fresnel to 0. Now if you render, you should get a completely opaque, mirror-reflective set of cubes.

2.59.5 Why Are My Shadows Black?

Consider the render at right: this is a default document with a plane added; the cube has ray-trace transparency turned on with an IOR of 1.5, Filter of 0.5, Depth of 5 and Alpha of 0, other settings left at their default. The plane has a material with all settings at their default.

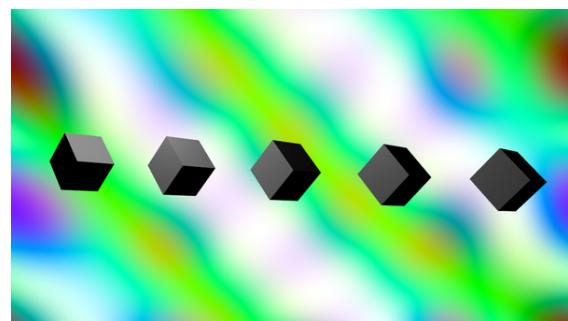
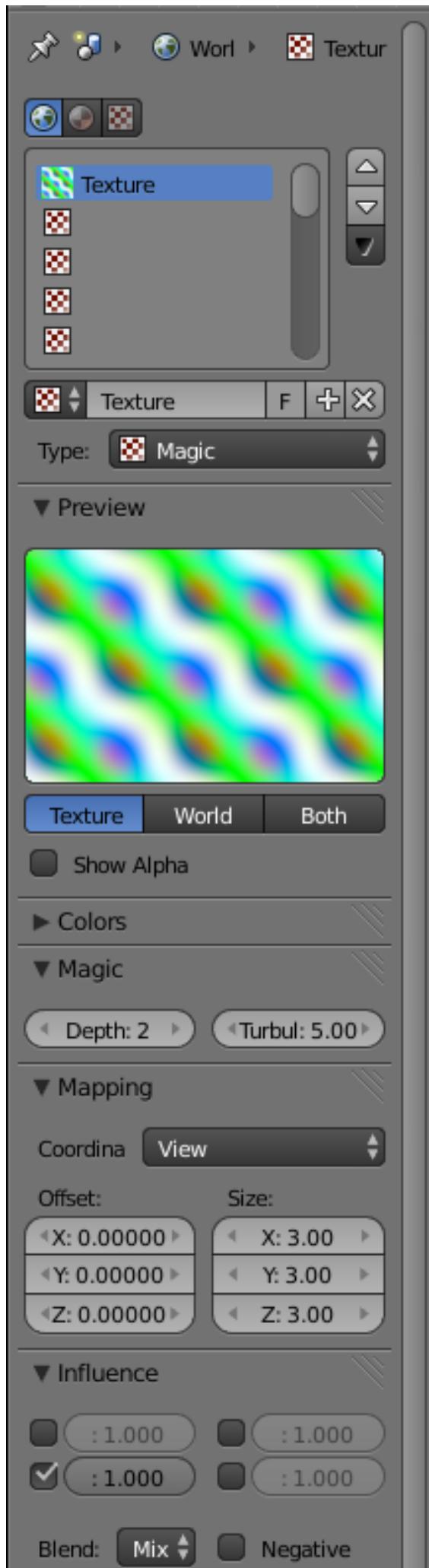
Light should be able to pass through the cube and illuminate the part of the plane directly behind; so why is the shadow completely black?

The problem is, by default, the Blender Internal renderer *doesn’t bother to compute transparent shadows*. Instead, it assumes that anything that blocks the light does so completely, presumably because it would slow things down too much to assume the opposite. To fix this, select the

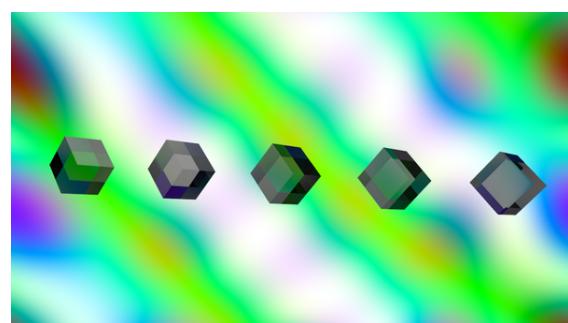
plane (*not* the cube), and in its Material  settings, look for the Shadow panel. Note the two checkboxes at upper left: by default the “Receive” one is checked, the “Receive Transparent” one is not. Check the latter as well. Now any object surface with this material assigned will have proper non-transparent shadows computed as appropriate.

Now when you re-render, you should see a more accurate-looking transparent shadow.

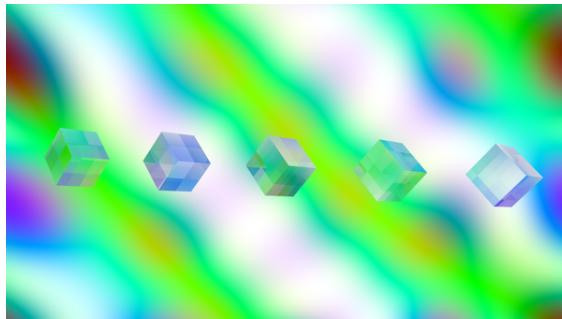
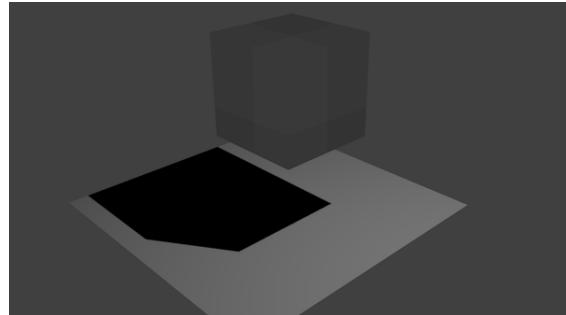
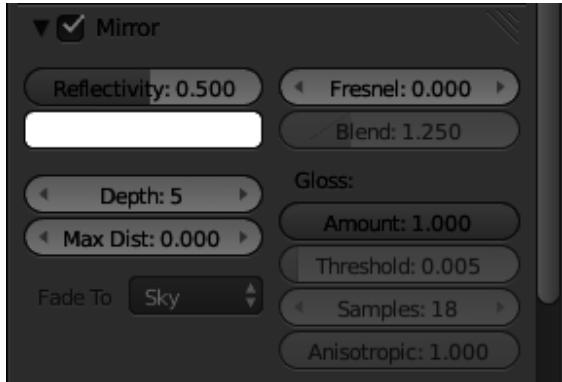
Imagine if you had a scene with lots of non-transparent objects with different materials receiving shadows from lots of transparent objects: you have to check this “Receive Transparent” option on *every single one* of those materials that might be receiving transparent shadows! Yes, this can be a pain. Not to mention the issue of *caustics*, which the BI renderer doesn’t handle at all. If you want to render realistic scenes like this, then you will need to learn about [Cycles](#)...



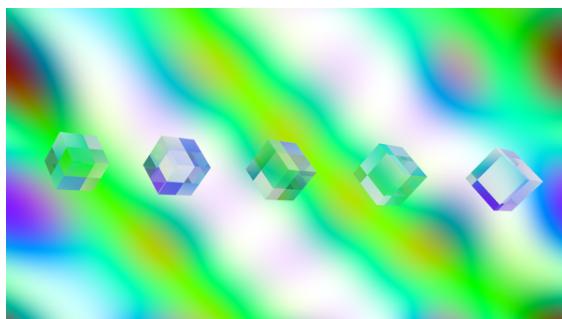
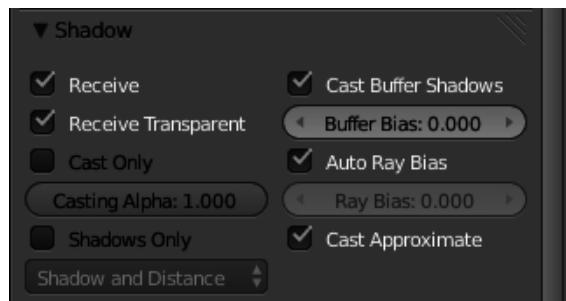
Left half of pool is filled with water material, right half is empty; floor is not broken, it just looks that way because the path of the light is bent.



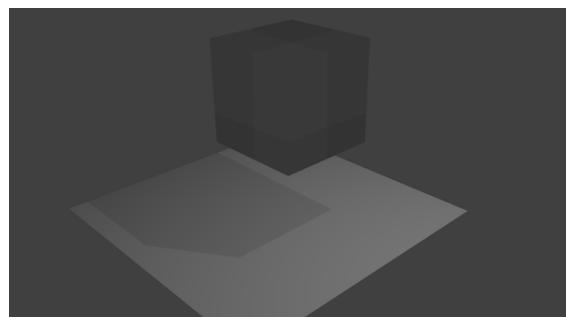
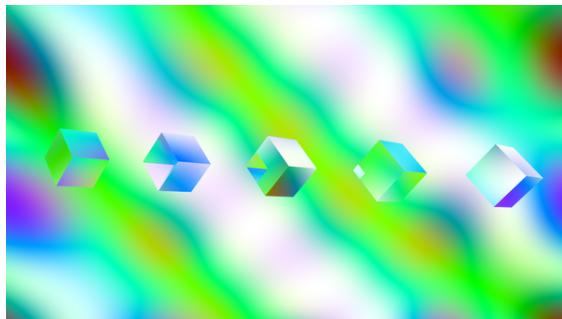
Semi-glossy...



Is that glass, or is that glass?



Even more glassy?



Pure mirror

2.60 Using Textures

Having said everything about Textures in Blender in the manual , many people find it still hard to grasp the concepts behind the texturing system and applying Textures.

There are some brilliant tutorials about texturing on the net, I personally like [Texturing for Dummies](#) , by Leigh Van Der Byl. It is free (as in free beer, not as in free speech like this book), so you should download and read it now, to take the most advantage out of this tutorial.

We're going to see how to apply the texturing concepts within the Blender texturing system. I will explain it using an image texture and procedural textures, but be aware of the fact that the texturing system is quite complex, and more advanced maps are created with the use of a few combined textures.

If you're really into texturing you need to learn how to [UV-Map](#), there's no way around it.

2.60.1 Colormap



Image 1a: An image applied as a texture to a plane. Download texture

This is the most basic mapping type. You can use an image or a procedural texture to change the color of a material.

Noob Note: A simple Basic texturing Guide [The Basics of texturing I](#).

Using an image texture:

- Add an *Image* texture in the *Texture Buttons* and load the image (this is necessary also for UV mapped textures).
- The button *Col* in the *Map To Panel* uses the RGB information of the image to change the color of the material.
- The slide number button *Col* governs the blending amount of the texture, *Neg* inverts the colourvalues.



Image 1b: The Map To Panel for a colormap.

- If you use *No RGB* only the intensity information from the picture is used. The target color is then taken from the RGB sliders in the *Map To Panel*.

Using a procedural texture: The result depends on the type of value the texture provides.

- If the value is of type intensity (e.g. *Clouds*), the color is taken from the RGB sliders in the *Map To Panel*.
- If the value type is color (e.g. *Magic*), everything is handled as with image textures.

2.60.2 Diffusemap



Image 2a: Diffusemap settings with a small range (0.2).

A Diffusemap changes the amount of diffuse light the material reflects/absorbs. This is controlled by changing the reflectivity (*Ref*). A material with a rough surface may have the color white, but absorbs light to a greater amount than a material with a smooth surface. For more on diffuse reflection physics, see [Diffuse reflection](#).

Noob note: Most of the settings described below will be found in the “Influence” section under the texture/material panel in blender 2.61

Let's test:

- In the Texture buttons (**F6**), in the Map Image panel, make sure *UseAlpha* is unchecked.

- In the Material buttons (**F5**), in the Map To panel, uncheck *Col* and check *Ref*.

Noob note: In blender 2.6x “Ref” has been changed to “Intensity”

To understand the effect of this and all other mappings, we have to discuss the meaning of the NumButton *DVar* (Destination Value).

- White in a texture (or an intensity value of 1) will be mapped to the value of the *DVar* Button. So if you set *Dvar* to 1, white will be mapped to 1, if you set *DVar* to 0, white will be mapped to 0. The result is multiplied with the value of the *Var* number button.
- Black on the other hand (or an intensity value of 0) does not change anything, so the settings from the material will stay unchanged. The difference between the material settings and the *DVar* button is the range for the texture.

Let's take a look at an example.



Image 2b: The image used for the Colormap applied as a Diffusemap with a small range (0.2).

If we use the default settings for the Diffusemap the range of the values is small.

- *Ref* is 0.8
- *Dvar* is 1.0
- White pixels in the texture change the *Ref* value to 1.0, black pixels in the texture don't change the *Ref* value at all.

So the resulting **range** for *Ref* is $1.0 - 0.8 = 0.2$.

When we change the *Ref* parameter in the *Shaders* panel to 0, we'll get a much larger range of *Ref* values from 0.0 (black pixels in the texture) to 1.0 (white pixels in the texture).

There are other ways to achieve a larger range, you could also set *Ref* to “Inverted” (click it twice, it will be painted in yellow), *DVar* to 0 and *Ref* in the *Shaders* panel to 1.



Image 2c: Diffusemap with full range (1.0).

2.60.3 Luminositymap

Luminosity is the property of self-illumination, i.e. of objects emitting light. So we change the *Emit* value with this texture.

This light does not illuminate other objects - you would have to use Global Illumination (e.g. Yafaray) to use an emitting object as a true light source. Radiosity rendering does not work with Luminositymaps.

Everything that was said about the *DVar* value applies here exactly as for Diffusemaps.

2.60.4 Specularitymap

Specularity is the second most important material attribute. Specularity fakes the reflections of light sources. Of course you can modify the general specularity with a texture. In Blender you can change three different attributes related to specularity:

1. *Spec*: The degree of specularity. You can't set values above 1.0 with Specularitymaps, but you could invert the texture and set *Spec* on the *Shaders* panel to a higher value and *Dvar* to the lowest value of your range. So the texture actually lowers the specularity in all areas, that shall have a low specularity.
2. *Hard*: The hardness of the specular reflections. Sometimes called “Glossiness”. A *DVar* of 1 is equivalent to a hardness of 130. Use the same method as described above to achieve a greater range for the hardness.
3. *Csp*: The specularity color.

You will probably use some kind of “Dirtmapping” (stenciling) to change the specularity in certain regions of an object, like an often touched object, that is shinier on the used or touched parts.

For more on specular reflection physics see Specular reflection.

2.60.5 Reflectionmap

A Reflectionmap would be used to fake real raytracing reflections, either because you don't want to use raytracing, or you don't want to create a scene, or you need a special effect you don't get with raytraced reflections. A Reflectionmap would be a Colormap, typically an Environment Map. Since you can use prerendered Environment Maps, you can fake a surrounding for your object. Sometimes it is sufficient to use a simple Colormap that just bears the right colours, e.g. if you want to create a stormy sea it is not necessary that the clouds above are truly reflected in the water.

Environment Maps are a bit more complicated to create, see the respective section in the manual.

A very different kind of texture would be used to change the amount of raytracing reflections, the *RayMir* value. This is useful for something like a stained or dirty mirror. Simply click the *RayMir* button in the *Map To* panel and lower the *DVar*.

2.60.6 Transparencymap

Transparency- or Alphamaps change the (partial) visibility of an object. They don't have to be transparent themselves (though it doesn't hurt, see [Make a material partially transparent](#)). You will often use an image as Colormap and Transparencymap together. A nice example for a Transparencymap is in the paragraph *Translucencymap*.

There's one catch though, you can't change Fresnel transparency with a Transparencymap. So you have to use plain transparency, or fake the Fresnel effect with the method described in the section [Map Input](#).

2.60.7 Refractionmap

Well, there's no such thing as a Refractionmap. You can't change the IOR with a texture. You may try and use Environment Maps to fake refraction, but the result is often not worth the effort (for stills, in an animation it might not be that noticeable).

2.60.8 Translucencymap

Translucency is a material property of all semitransparent materials, like frosted glass, paper, plastic, cloth, skin, stained glass and the like. It allows objects to be lit from behind.

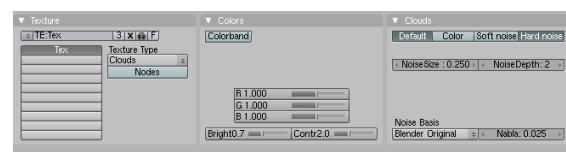
Totally clear glass does not show where the lightrays travel through it. But if the glass is dirty or uneven you will see the path of the light. In the first example (*Translucency 1*) the glass material has a transparency of about 0.5 and a translucency of 0.78. So the glass is brighter where

the lightrays hit the surface from behind. Additionally a Transparencymap - which sets Alpha to 1 - lets the surface appear to be stained. If we use the same map also as a Translucencymap (*Translucency 2*), the stain appears to be lit - which it would be of course also in reality.



Material settings for example Translucency 2 (blue glass).

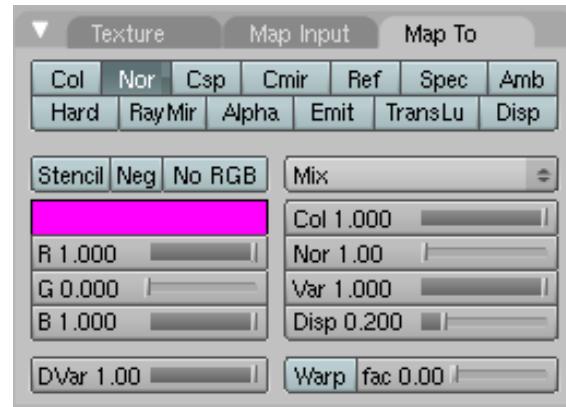
The settings for the material (*Z-Transp*, but raytracing shadows):



Adjusted Clouds texture.

The texture itself is a *Hard Noise Clouds* texture, with modified *Brightness* and *Contrast*.

2.60.9 Bumpmap/Normalmap



Settings for a Bumpmap in the Map To panel.

Bumpmaps are a technique to create the illusion of geometry. They could be used for something like a canvas or any small structures that are either difficult to model, or are too computationally expensive. As long as you don't get too close with the camera you won't notice the difference.

They come in several different flavors:

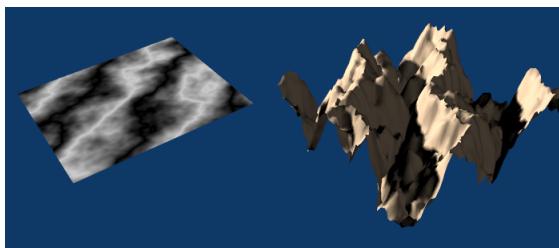
- A Bumpmap texture in the nearer sense is a greyscale image or procedural texture mapped to *Nor*. Don't activate the button *Normal Map* in the *Texture Buttons* for a Bumpmap.

- A Normalmap is an RGB image whose color information bears the information about the direction of the surface normal, also mapped to *Nor*. For this you need to activate the button *Normal Map* in the *Texture Buttons*.

See also the manual about the differences between **Bump** and **Normal Maps**.

In the example the bump map was used additionally to the colormap. The *Nor* slider in the *Map To* panel sets the depth of the bumping.

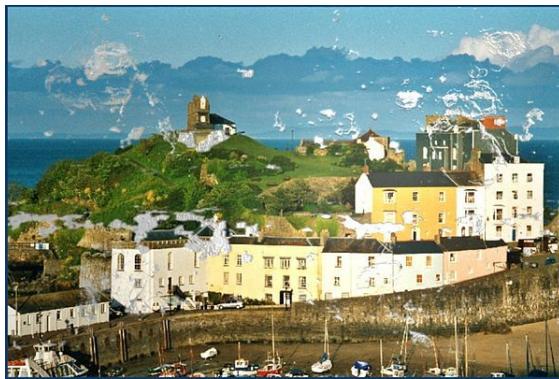
2.60.10 Displacement Map



A displacement map deforms the mesh

A Displacement Map creates real geometry, i.e. it moves the vertices of a mesh. Therefore you need a lot of vertices. You can use it as a texture mapped to *Disp*, or even more flexible using the **Displacement Modifier**.

2.60.11 Dirtmap



Our picture has seen some bad times.

I will try to explain what a Dirtmap is and how you apply one. There are different things that people mean when they talk about dirtmapping.

- Sometimes a Diffusemap is meant.
- Dirtshading is often used as a term for Ambient Occlusion.

With a Dirtmap I simply mean any texture that breaks the clean, uniform, untouched look of image or procedural textures. You may do this by simply combining some partially transparent textures resp. textures that affect only parts of the picture in a semirandom way. You can also select parts of the object with a *stencil* texture. And of course you can (and will) combine both of these methods.

As long as two textures don't affect the same property you can simply apply one after the other. To select only parts of your texture to apply another texture with the same properties use stenciling. Or use stenciling as a convenient method to separate two different sections on your texture.

You need at least three textures for this.

1. The basic texture (e.g. color).
2. The selecting (stenciling) texture.
3. The Dirtmap affecting the same or other properties than the first texture.

The stenciling texture has to provide an intensity value, so if you want to use an image (or other RGB) texture you must activate *No RGB*.



The stenciling texture.

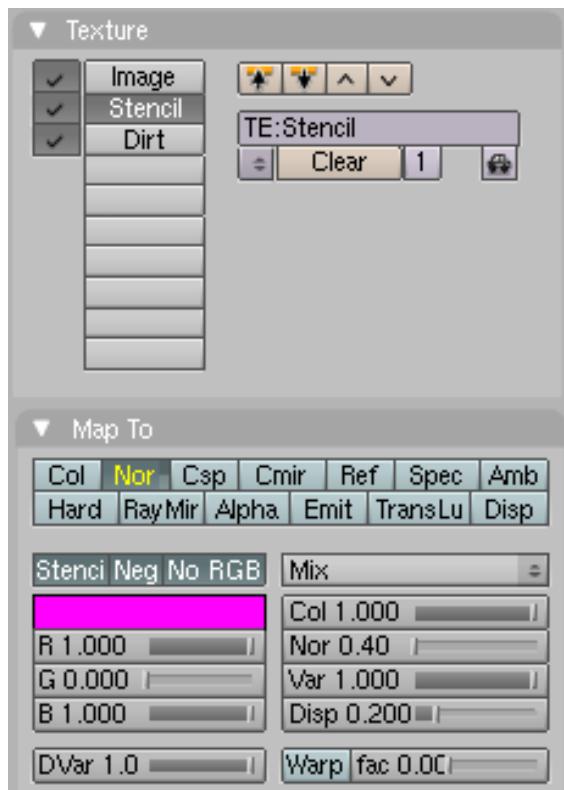
The image texture used as a dirtmap for our example contains some semirandom noise.

Nor for the stenciling texture was not really necessary, but it adds a nice 3D effect. Since the stenciling texture is mostly white, I had to use *Neg* to use the black pixels as mask.

The third texture ("Dirt") affects *Col*, *Nor* and *Spec*.

2.60.12 Use UV Coordinates for your Maps

It is often necessary to create UV coordinates for more complicated textures/objects. UV coordinates only describe the geometry of the mapping, i.e. which pixels to



Settings for the stenciling texture.

map on what face. You can use the UV coordinates to create any mapping you want.

1. Create an UV map for your object. This is beyond the scope of *this* tutorial, see [UV Map Basics](#) and the section about UV Mapping in the manual.
2. Don't turn *TexFace* in the *Material Panel* in the *Material buttons* on. Though *TexFace* is a very simple way to use UV textures, the normal texture mapping methods are much more flexible.
3. Load the image you used for UV-mapping (or any image with the same size) as an image texture. Change *Map Input* to *UV*.
4. Now you can use this image - or multiple different images - for your maps, including color-mapping.

The advantage of this technique is that you have much better control over your texture. Even if you just want to texture something like a cube shaped object, you're often better off with UV mapped textures.

2.61 Using a texture to make a material partially transparent

To understand the collaboration of textures and material we will use the opacity of a material - the alpha value

- as an example. We will show how to set the opacity/transparency of a material with different types of textures.

The Alpha value of the material is set with the *A(Alpha)* slider in the Material panel. This is the basic Alpha value. If *Alpha* is 1, the material is fully opaque, if *Alpha* is 0, the material is fully transparent.

Now you change that basic Alpha value with a texture.

1. If the basic Alpha value is 1 you can only lower it with a texture, because 1 is the maximum value anyway.
2. If the basic Alpha value is 0 you can only increase it with a texture.

The target value of Alpha that can be achieved with the texture is set with the **DVar** button in the *Map To* panel. If *Alpha* is 0 and *DVar* is 0 than nothing will happen. You can set either:

1. *Alpha* to 1 and *DVar* to 0, then the texture will lower the Alpha value, or you can set
2. *Alpha* to 0 and *DVar* to 1, then the texture will increase the Alpha value.

Both methods are possible and equal in the result, but for clarity we will always use method number 1. There are more combinations possible, so it may be that you find tutorials with different settings.

2.61.1 Using a greyscale texture - white for opaque



Image 1a: Greyscale Image of leaf

In **Img. 1a** the black areas shall become transparent, the white areas shall become opaque.

- Load the image as *Image* texture, turn off *Use Alpha*.
- Set the *Alpha* value in the material buttons to 1. This is the default value anyway.
- Activate *ZTransp* (or *RayTransp* if you need refraction)
- The *Map To* panel:
 - Turn off *Col* (or else the texture would also affect the color of the material)
 - Turn on *Alpha* inverse, you have to click it twice. We need inverse here, because the texture shall affect the material from its black parts. (**Noob Note:** *On blender 2.63 you need to set a specific 'Alpha' value (the one located under 'influence->diffuse) to -1, as the alpha button referenced in this step no longer exists in 2.63.*)
 - Turn on *Spec* inverse (you don't want your transparent sections to show specular highlights) (**Noob Note:** *As far as I could tell, the equivalent to doing this on blender 2.63 is to set a specific 'Intensity' value (located under Influence->Specular) to -1.)*
 - Set the *DVar* value to 0. The default value is 1.

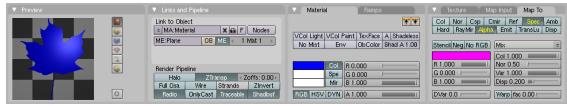


Image 1b: Material settings for the texture

Now let's take a look at the texture.

- Where the texture has a value of 0 (e.g. black) it will affect the Map To values fully (because we have used inverse).
- Where the texture has a value of 1 (e.g. white) it will do nothing.
- Where the texture has a value in between 0 and 1 it will affect the Map To values partially.

The result is shown in **Img. 1c**, the texture applied to a plane rendered against a white world background.

If your texture is inverted, i.e. black shall become opaque and white transparent, simply use the normal *Alpha* and *Spec* setting in the *Map To* panel. (**Noob Note** *If you don't see the white 'shine' on the maple leaf, move the default point lamp (lighting source) closer to the plane mesh to which the texture and material are assigned.*)

That's it. It's basically the same for all settings, the only problem that remains is sometimes to know which value



Image 1c: Result of applying the Greyscale Image to a plane

a texture will give if you use transparent textures or colored textures.

For 2.7 blenderers out there: To obtain the results I had a hard time and finally came across (almost accidentally) to uncheck the 'use alpha' button in the image sub panel of the texture menu. Then, you can get the results as above.

2.61.2 Using a partially transparent texture to make the material transparent



Image 2a: Using a partially transparent texture

Often we have a partially transparent texture, and want to directly use that transparency of the texture to make the material transparent too. We have now to think a bit

about the values that are generated by the texture itself: If we load the image texture with *Use Alpha*, the Alpha value of the image is used as input value for *Alpha* and *Spec* in the *Map To* panel. So a fully transparent pixel in the image has an Alpha value of 0, a fully opaque pixel has an Alpha value of 1. This is exactly the same situation as above, simply turning on *Use Alpha* and *Premul*(tippy Alpha in Advance) will give use the same result as with the greyscale image.

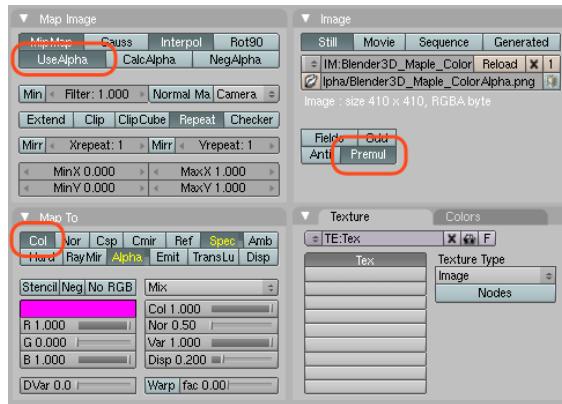


Image 2b: Texture settings for a RGB/Alpha texture

So:

- Turn on *Use Alpha* in the *Map Image* panel of the texture buttons.
- Turn on *Premul* in the *Image* panel. This is necessary to get real partial transparent borders.
- Turn on *Col* in the *Map To* panel additionally.

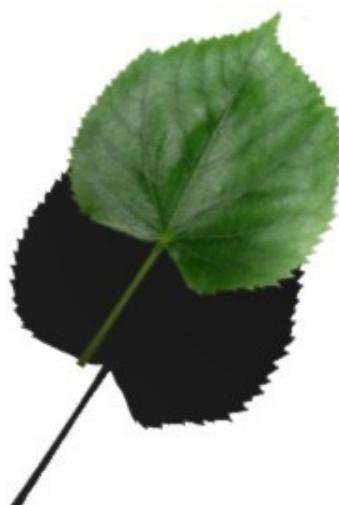


Image 2c: The result

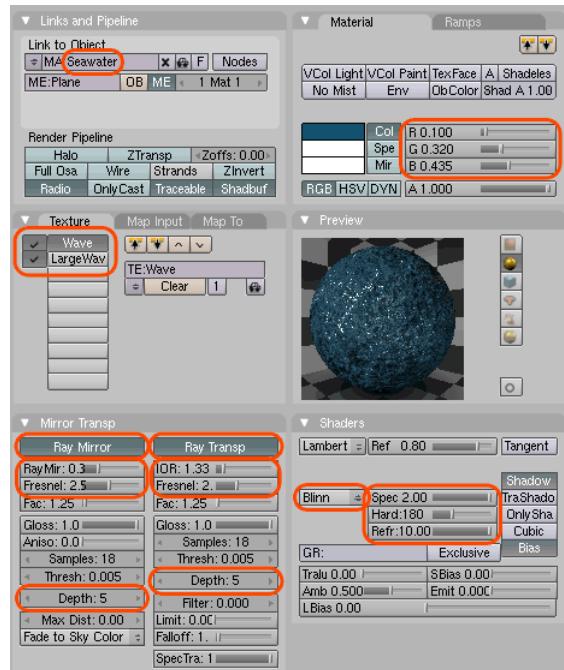
The rendering result is pretty similar to **Img. 2a**, only that it is affected by lighting (and casts shadows).

Don't forget to turn on *TraShadow* in the *Shaders* panel for the object that is **receiving** the shadow. (**Noob Note:** In Blender 2.6x you also need to check the "Receive Transparent" box in the shadow section of the material for the object receiving shadows or you will get a black square shadow.)

(**Noob Note:** In this case add a plane colored white just below the "leaf." This is the object the you should turn on **TraShadow** with as this is the object that will receive the shadow. This is also providing that you have a light source properly positioned.)

2.62 Creating Basic Seawater

[ed. note: Need a much more basic introduction to what materials, textures, maps, and all the accompanying terms are with illustrative examples before diving into a specific sea-water example. Much more effective learning when you know what you're changing.]



Settings for the seawater material

75% of the Earth's surface is covered with water. In homage to this great fact, we will develop your materials skills first by creating basic seawater.

2.62.1 Create 3 linked planes

First we create a new file in Blender and delete the default cube by pressing **XKEY** and confirming the popup dialog. Now switch to top view with **NUM7** and enter

SPACE > Add > Mesh > Plane to create a plane. Then scale it up to 20 its original size with the **SKEY** the way you've already learned in one of the earlier tutorials. Go to the side view with **NUM3** and duplicate this plane *two more times* using **Alt-D** (not Shift-D), moving the plane down on the Z axis by two grid spaces each duplicate. This will make the transparency of the water more realistic once we set it. Using Alt-D rather than Shift-D makes a linked duplicate, so that the changes we make to one plane effect the other two.

2.62.2 Create material

Now off to the actual texturing work. Select any one of the planes and press **F5** to bring up the Material Buttons in the Buttons Window. You will probably find two new small windows appearing here: one called Links and Pipelines and the other one Preview.

(**Noob note:** A new section has been added for Blender 2.63 users after this section.)

Click the 'Add New' button in the 'Links and Pipelines' tab to create a new material named 'Material.001' or so. To make life easier we'll rename it to something meaningful like 'Seawater' by simply clicking it and typing in the letters, as shown here (SHIFT+DELETE in field to clear):

Now, on the same tab, give the seawater material a color of RGB (0.100, 0.310, 0.435). Find the tab that reads 'Mirror Transp' and click it. Click on 'Ray Mirror' and 'Ray Transp'. For the "Ray Mirror" box, move the 'RayMir' slider to 0.3, the 'Fresnel' slider to 2.5 and the Depth to 5. For the 'Ray Transp' box, move the 'IOR' slider to 1.33, the 'Fresnel' slider to 2.0 and the Depth to 5. This will give the water realistic transparency and reflection. Also click the 'Shaders' tab, change 'CookTorr' to 'Blinn', move 'Spec' to 2.000, 'Hard' to 180 and 'Refr' to 10.000. This will make the water look more glossy.

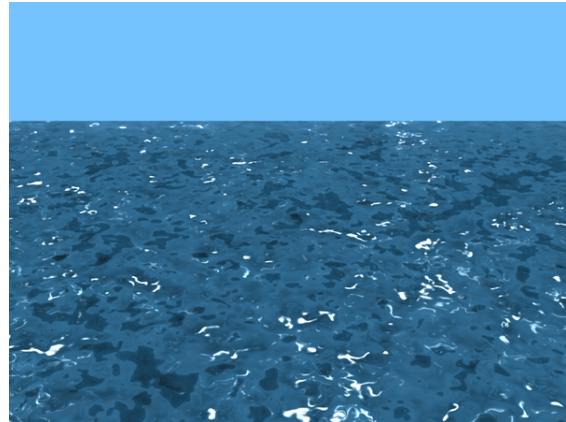
Now we'll add a procedural texture to our seawater, which will give it a "wavy" look. Click the Texture button (looks like bricks) or press **F6** to view the texture buttons sub-context. Click on the knob to the left of the texture name and select the "Add New" button. This creates a new texture named "Tex.001" or so. Click on the name and change it to "Waves".

Go to the Texture Type pull-down (**F6**) and select 'Clouds'. On the Clouds tab change 'NoiseSize' to 0.050. Our Waves texture is ready; next, we will refine how it is applied to our Seawater material.

Noise Size increases the size of the noise, in this case, the clouds. Soft Noise blends the intensities and reduces the contrast. Makes a mellow effect, like soft waves. Hard noise creates a high contrast, and brings out individual 'shapes'.

If you want to add more detail to your water, add another

texture and rename it to "LargeWaves". Make it a cloud texture like the previous one, but make it's 'NoiseSize' 0.300 and use 'Hard noise'.



The final rendered image

Left click on the Materials button (looks like a red sphere) to return to the material buttons subcontext. Look at the Texture panel, and you'll see that the "Waves" texture has been automatically associated with the Seawater material.

Select the 'Map To' tab. Click the 'Nor' and 'Spec' buttons so they're selected and have white text (the white text indicates a positive mapping). Click the 'Hard' button twice so it's selected and has yellow text (the yellow text indicates a negative mapping). Click the 'Col' button so it is not selected, this button will show any color in the texture which we do not want. Find the 'Nor' Slider and move its value to about 5.00.

If you created the "LargeWaves" texture, select the "LargeWaves" texture under 'Texture and Input', go to the 'MapTo' tab, deselect 'Col', select 'Nor' and move the 'Nor' slider to 7.00. Do not select 'Hard' or 'Spec' this time.

For lighting press **Space > Add > Lamp > Sun**. You shouldn't need to move the sun or change any of its settings. Finally move the camera to the edge of the plain and move it up towards the sky a bit.

Go to the Scene tab (**F10**), and look for the six buttons next to the big render button. Deselect all these, leaving only the 'Ray' button selected. This will tell Blender not to render some features in our scene that we really don't need. Go and press **F12** to render the water, it may take a while depending on your system.

Admire your water, and maybe drink a tall glass of something refreshing!

2.62.3 Create Material / Textures (Blender 2.63)

Material

- Go to the properties window (Bottom right window

- by default) and click on the Material tab
- Click of the + symbol to create a new material and name it Seawater
 - Click on the diffuse color box and give it a seawater blue RGB (0.100, 0.310, 0.435)
 - Enable the Mirror checkbox
 - In the Mirror section, set Reflectivity: 0.3, Fresnel: 2.5 and Depth: 5
 - Enable the Transparency checkbox
 - In the Transparency section, click the Raytrace button
 - Also in the Transparency section, set IOR: 1.33, Fresnel: 2 and Depth: 5
 - In the Specular section, use the pull-down menu to change CookTorr to Blinn, then set Intensity: 1, Hardness: 180 and IOR: 10

Textures

- Still in the Properties window, click on the Textures tab
- Click on the “+ New” button to create a new texture and name it Waves
- Set the Type: Clouds using the pull-down menu
- In the Clouds section, set Size: 0.05 to create soft noise
- In the Influence section, uncheck Color under Diffuse, then under Specular, check the “Specular” box and “Hardness” box then set the value to Hardness: -1. Under Geometry, check the “Normal” box and set the value to Normal: 5

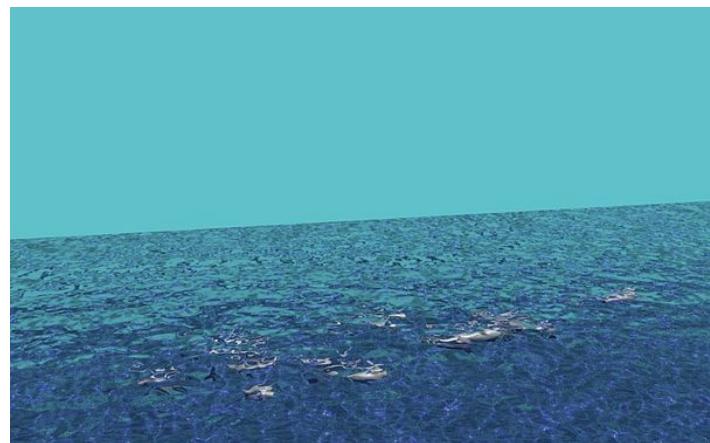
(**Noob Note:** V. 2.78 Under Mapping change coordinates to Generated.)

- Select the second line in the Textures list and create a second Texture using the “+ New” button. Name it LargeWaves
- Set the Type to clouds again, but this time, set Size: 0.3 in the Clouds section and click on the “Hard” button for hard noise
- Deselect “Color” and under Geometry, select the “Normal” box and set the value to Normal: 7

(**Noob Note:** V. 2.78 Under Mapping change coordinates to Generated.)

Lighting

- In object mode, select the default lamp (if present) and delete it
- Create a new lamp by pressing ALT+A and selecting Lamp --> Sun
- Place the Sun in the sky by moving it up the Z-axis. Press G, Z, 20 and hit ENTER
- Go back to the Properties window and click on the World tab
- Click on the box under Horizon Color and set the color to RGB (0.242, 0.617, 0.831) for sky blue
- Position the camera to be looking over the water with some sky visible and render your scene.
- How about using what we have learned in previous modules to add some dolphins swimming just below the surface?



2.62.4 Extra Practice

This tutorial might also help you make even more realistic water: [Link](#)

2.63 Mountains Out Of Molehills 2

This page is broken, minimal required information is missing.

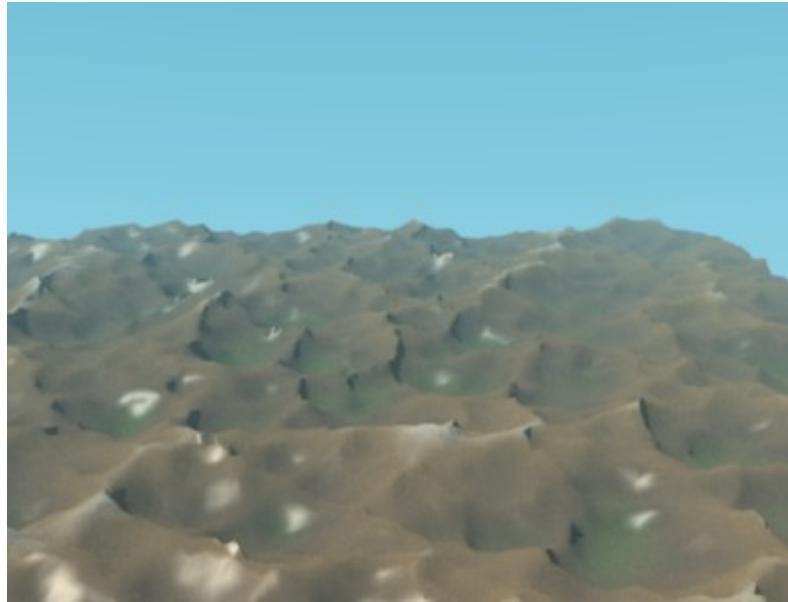
This tutorial shows you how to use displacement mapping to make a simple environment.

- Make a grid. (Add/Mesh/Grid) 32x32 will do just fine. Be sure to create a Grid instead of a Plane, or you'll end up with just a flat plane! This is because a Plane only has four vertices to manipulate - one for each corner - while a Grid has many, many more. A 32x32 Grid, for example, would have over 1000 vertices. The more vertices there are to manipulate, the more effect displacement mapping will have on the finished product.

2. Set it smooth. (Editing/Link and Materials/Set Smooth)
3. Make a new material for it. (Shading/Material/Add New)
4. Make a new texture for the material. (Shading/Texture/Add New)
5. Go to Shading/Texture Buttons. You can see your newly created texture there now.
6. Change Texture Type to Clouds.
7. Change the name of the texture to be more descriptive. For example GroundDisp or something similar.
8. Go back to Shading/Material buttons. You can see our cloud texture applied now, but it's not applied correctly yet. Let's fix this next.
9. Go to Shading/Map To. This defines how the selected texture is mapped on our material. Uncheck Col, check Disp on, and set the Disp slider to a value like 0.200
10. Set camera and a few lights to the scene. (This is already done as part of the default scene in recent versions of Blender, such as 2.49)
11. Render.

Noob Note It looks like from version 2.70 there is no shading as a main tool but rather as a subtool for Material, so in version 2.70 to get the effect described in here you have to check third checkbox in Texture>Influence>Geometry and adjust the slider

Several individuals, when working through this tutorial, had trouble getting anything more than a flat plane. A few solutions were proposed (most are still visible on the Talk page), and while each worked in its own way, the vast majority of the time the problem stemmed from creating a Plane instead of a Grid. Be sure to follow each step very carefully to ensure you don't miss anything or do a step incorrectly.



You can tweak the environment easily by changing Nor value in the Shading/Map To. This defines how strongly the displacement texture affects the material.

You could also add subsurfing to the ground area to get smoother results. Also feel free to tweak the texture and try out different alternatives.

Once your mountain looks good, try adding some Mist.

1. Select the "Mist" button on the "Mist/Stars/Physics" Tab among the World buttons

2. Add a Cloud Texture to the World and make it blend from white to gray

The end result is something like this:



The purpose of this tutorial, is to highlight the power of blenders built in shaders and procedural textures to create a carpet material to use in your scenes.

NOTE: For those of you needing help getting a similar scene to the one above, here are some axis positions, etc. to help out:

- (Spot-Lamp)-X=1.62,Y=0.86,Z=6.74;rotation-X=37.26,Y=3.16,Z=181.34; (Area-Lamp-1)X=4,Y=3.27,Z=4.12;rotation-X=54.67,Y=-18.59,Z=-109.47; (Area-Lamp-2)X=-2.07,Y=-2.08,Z=4.85;rotation-X=29.37,Y=-28.98,Z=355; (Area-Lamp-3)X=0.315,Y=-2.89,Z=4.29;rotation-X=49.23,Y=-10.63,Z=6.68; (Monkey)X=0.05,Y=0,Z=0.42;rotation-X=58.61,Y=-16.07,Z=23.245;DIM(dimensions)X=2.734,Y=1.969,Z=1.7 (Plane)x=0,y=0,z=0;(no rotation);Dimensions: X&Y=14.30,z=0.

Alternatively, you could also download [a pre-made file](#) in case that you're in a hurry.

Noob Note: Try adjusting each of the area lamps Dist(ance) value.

Noob Question: How does the above coordinates help me at all? I haven't found anything that tells me where in 3D space object actually is or the objects dimensions little alone control them. Nothing in the Tutorial up to now even gives a clue to this.

Noob Answer: Hit 'N' to bring up the Transform Properties window and enter the values there.

2.64.2 The Basic Material and Shader Settings

2.64 Basic Carpet Texture

2.64.1 Goal

I am using a basic scene that I quickly set up before I started to create the carpet material. It shows a monkey (suzanne), a plane, camera, 3 area lamps and 1 spot lamp.

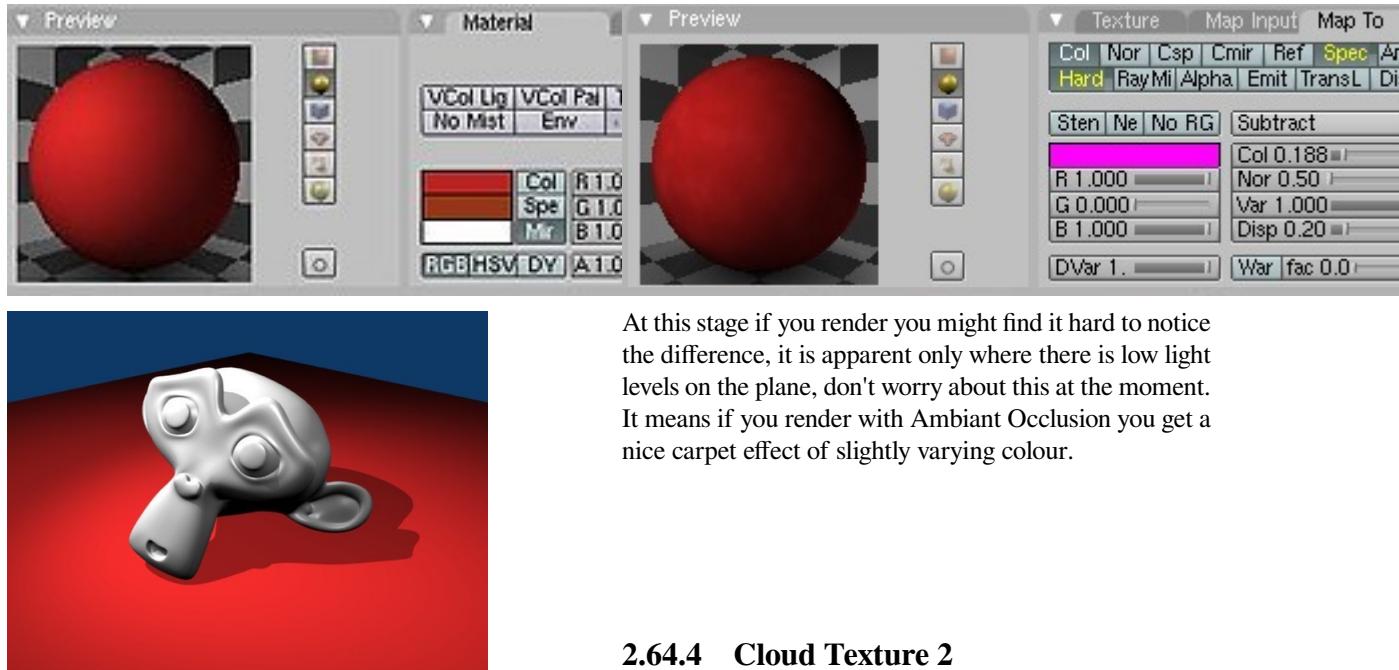
For the basic material for the carpet set the color settings of your material as follows -

Col (R 0.714) (G 0.134) (B 0.134) Dark Red
Spe (R 0.590) (G 0.210) (B 0.084) Reddish Brown
Mir (R 1.000) (G 1.000) (B 1.000) White

Change the specular shader to ('phong') and leave the default lambert diffuse shader as it is. Change the ('spec') to [0.13] and the ('hard') to [12]. Lastly click the ('Full Osa') button to enable it. If you render now you

will notice that the plane looks like an ugly pastel color (if not, you have a different lighting setup to mine and the shader will not look the same as the images in this tutorial.) Don't worry about this ugly looking plane it will soon be a beautiful carpet.

Now go back into the material settings and change the settings in the Map To tab as follows. Click ('Spec') twice so the text becomes yellow do the same for ('Hard'). Now select 'Subtract' for the texture blending mode. Change ('Col') to [0.188].



At this stage if you render you might find it hard to notice the difference, it is apparent only where there is low light levels on the plane, don't worry about this at the moment. It means if you render with Ambiant Occlusion you get a nice carpet effect of slightly varying colour.

2.64.4 Cloud Texture 2

2.64.3 Cloud Texture 1

Press 'F6' on the keyboard to bring up the texture panel. Click the bottommost of the long boxes to create a texture in the bottom channel. Create a new texture and rename it something like 'Red Clouds 1'. From the ('Texture Type') pull down select clouds. In the 'Clouds' settings panel change ('NoiseSize') to [0.210] and ('NoiseDepth') to [4].

Now select the Colors tab which will bring up the ColorBand for the texture. Press the orange ('Add') button to add a cursor on the colourband. Next make sure the 'Cur : 0' is showing next to the add button and change the colours as follows - (R 0.770, G 0.168, 0.168). Now click on right side of the 'Cur : 0' so it shows 'Cur : 1'. Change 'Pos' to 0.6. Set Alpha to 1 and change the colour to (R 0.732 G 0.243 B 0.243).

Go back to the texture panel and create another texture, call it 'Clouds' and put it in the channel above 'Red Clouds'. Change ('NoiseSize') to [0.054] and ('NoiseDepth') to 4. Select 'Improved Perlin' from the Noise Basis pull down. Finally change ('Nabla') to [0.031]. Do not change any more settings here.



Now in the material panel, under the Map To tab change the following - Click ('Nor'). Click ('Spec') and ('Hard') twice so as they are yellow.





As you can see the material is starting to look a bit better, only 2 more textures to go.

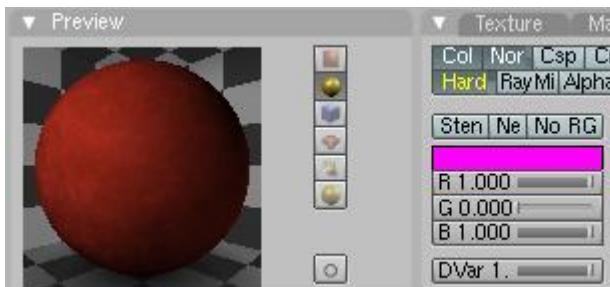
2.64.6 The Last Texture

2.64.5 The Final Cloud Texture

Switch to the texture panel once again and create a new Clouds texture in the next channel up. Change ('NoiseSize') to [0.010] and ('NoiseDepth') to [6]. Now click on the 'Colors' tab and change the colour of the left cursor ('Cur : 0') to (R 0.713 G 0.262 B 0.223) and Alpha to 0. Switch to Cursor 1 ('Cur : 1') and its colour settings (R 1.000 G 0.363 B 0.000) and Alpha to 1.



Now in the materials panel under the 'Map To' tab Click ('Nor'). Click ('Spec') and ('Hard') twice so as they are yellow. Leave the blending mode as 'Mix'. Change ('Col') to [0.464] and ('Nor') to [1.00].

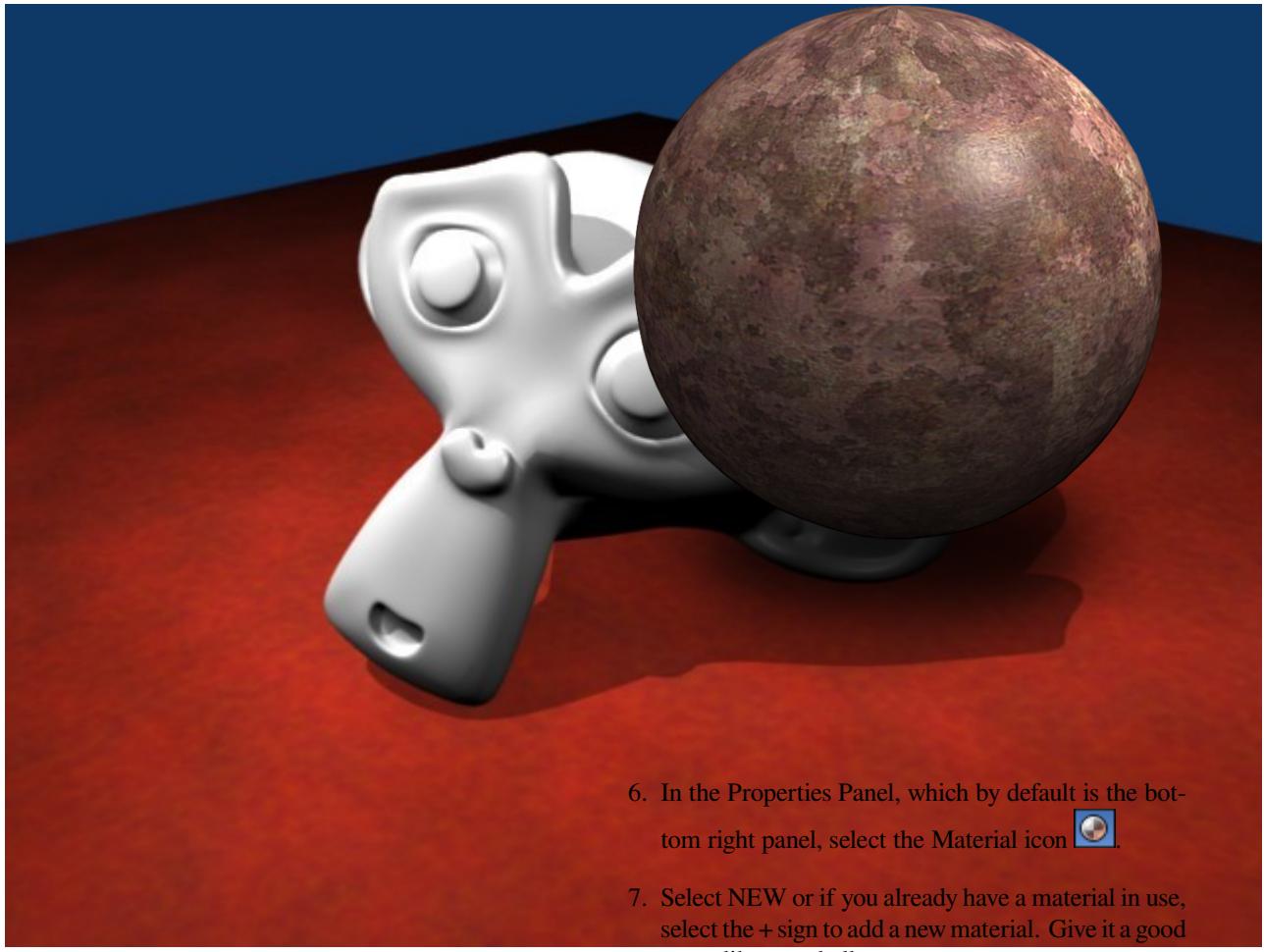


Finally, go to the textures panel one last time. Create a new Stucci texture in the next channel up. Change ('NoiseSize') to [0.006] and ('Turbulence') to [10.94]. Click on the 'Colors' tab. The first cursor 'Cur : 0' should be black with alpha 0. 'Cur : 1' should be red (R 1.000 G 0.000 B 0.000) with alpha 1.



Now go to the materials panel. Under the 'Map To' tab Click ('Nor'). Click ('Spec') and ('Hard') twice so as they are yellow. Leave the blending mode as 'Mix'. Change ('Col') to [0.056] and ('Nor') to [0.50]. And that's it. When you render now, you should have a nice-looking carpet material. By tweaking with the colours you can create any colour of carpet.





2.65 The Rusty Ball

Making objects with image textures is not really hard for simple objects like balls, cubes, and tubes. I'll show you how to do this:

1. Make a new scene in Blender and delete the default cube.
2. Add a sphere. Apply a Subsurf modifier and Smooth as we have learned in previous tutorials.
3. Find a file picture that you want to apply to your object. Check Flickr for something like "rust texture." (If you are going to share it, be sure you check the license of the texture you download.)
4. **IMPORTANT:** Sometimes you need a seamless texture. If you are not sure, use a program such as GIMP. Use FILTERS > MAP > MAKE SEAMLESS. Usually this works well but if your texture turns into gorp, try another one.
5. **(Noob Note: Do a search for Rust Textures if you do not have one handy.)**

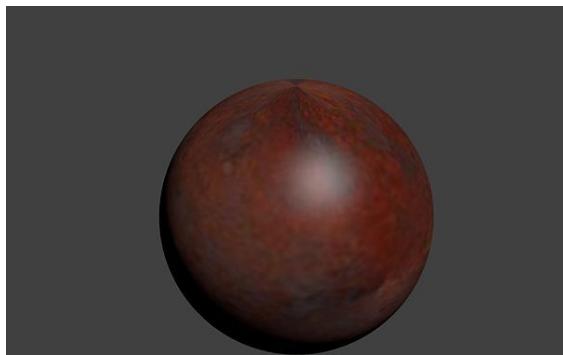
6. In the Properties Panel, which by default is the bottom right panel, select the Material icon
7. Select NEW or if you already have a material in use, select the + sign to add a new material. Give it a good name like rustyball.
8. Click on the Texture Icon
9. Click NEW or if you already have a texture loaded, click on the + to add a new one. Give it a good name like RUST.
10. The default type is CLOUDS. Pull down the menu and change it to IMAGE OR MOVIE.
11. Go to the IMAGE section, choose OPEN. New options will become available. (Note: JPGs, PNGs or TGAs are recommended for Blender. Bitmaps tend to get all screwy.)
12. Under Source: Click the folder icon and navigate to your image.
13. Scroll down. In the Image Mapping area are some options that you may need to try. If your image is small it may have to be repeated in the X or Y directions.
14. In the Mapping section (yes, one section is Image Mapping and the other is just Mapping) after Projection, choose the shape closest to your object. If you are using a sphere like the example, then choose ... you guessed it ... SPHERE.

- The Preview often looks a bit odd and stripey, but render your project and it should look very nice.
- This method works for all sorts of things. Try making a brick wall with a wooden gate using Flickr textures or go back to the Jeep tutorial and put a pattern on your jeep.
- You can also render videos onto objects using this method. Just select a movie in the “Load image” dialog and enable the option “Movie” at the textures buttons. *NOTE:* Blender ONLY works with Full Resolution video, not video which has been compressed using a codec. Most video software will allow you to export video as “**full frames**” or “**no compression**”. Experiment a bit!

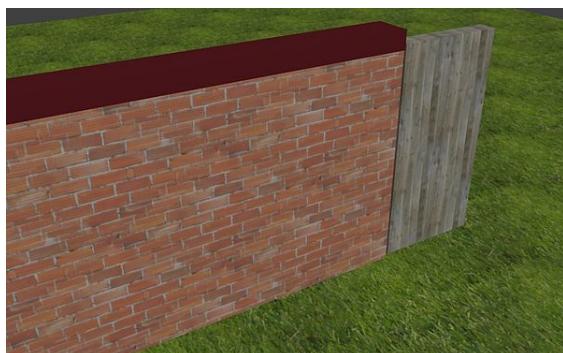


Psychedelic Jeep, my own texture such as it may be

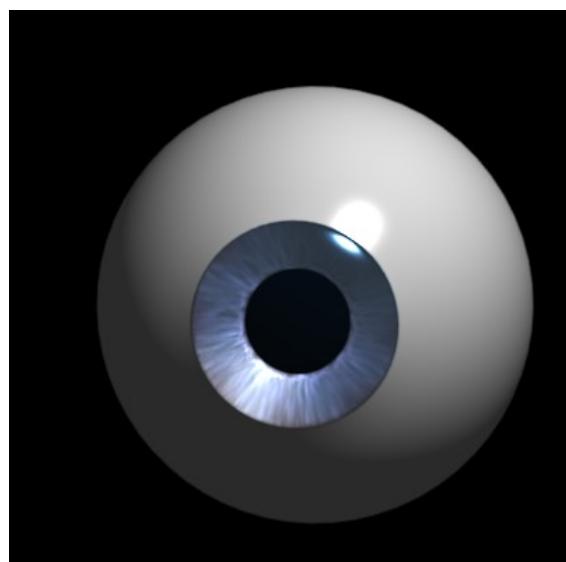
2.66 Creating Pixar-looking eyes



The Rusty Ball



Textures used from Flickr under Creative Commons 2.0 license



The final result, with a blue iris.

Note: This tutorial uses the same modelling and texturing technique described in the well-known MAX tutorial by Adam Baroody (<http://www.3dluvr.com/rogueldr/tutorials/eye/eyes.html>). The sole purpose of my tutorial is to make this technique more popular among the Blender users by explaining how to achieve the same result with Blender.

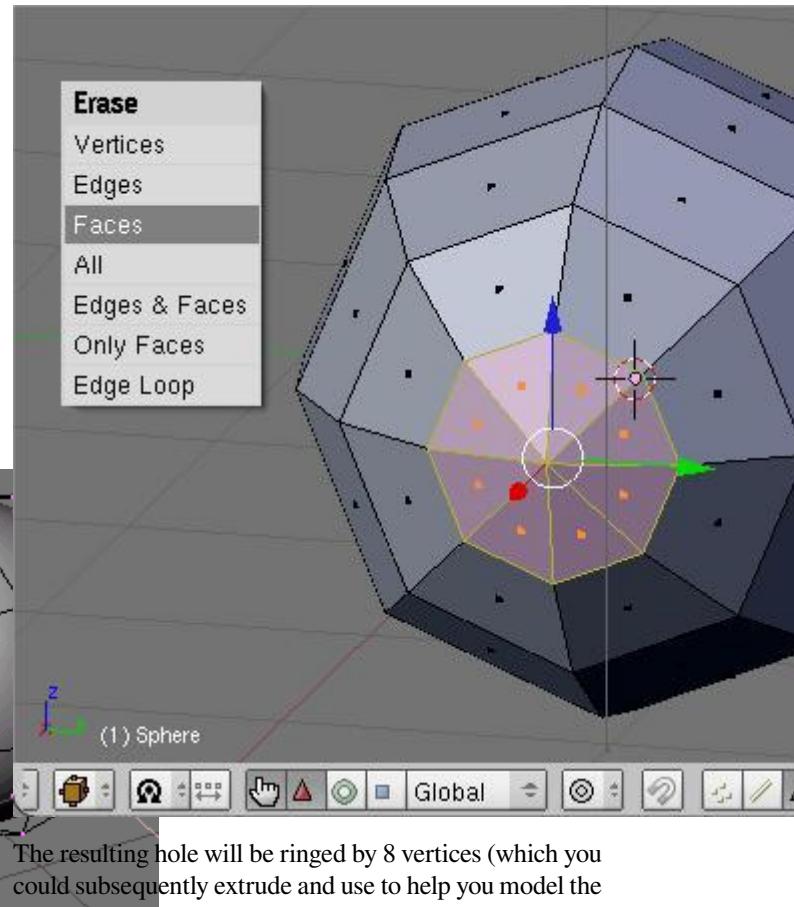
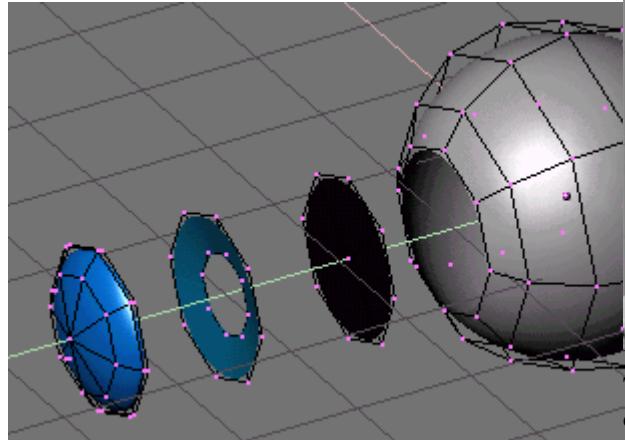
The goal of this tutorial is to make a Pixar-looking eye. One of the main reasons that Pixar’s characters really convey life is in their eyes. They have depth, you can see how the eye not only shines but it “collects” light. You may think that you can’t achieve this effect without raytracing but you’re wrong. The secret of this depth is in the modelling of the eye. Let’s see how it works!

2.66.1 Parts of the Eye

In this picture you can see the “ingredients” of the eye model. The blue mesh at the left is the cornea. Its shape

allows for a small spot of specular light to appear on it even if the light is in a far side position. The mesh next to it is the iris. Now notice how it's a bit concave. That's the tricky part - the shape of the iris allows for a wide soft specular light to appear at the opposite side of the lamp direction. This fakes refracted light from the cornea and makes the illusion of "collecting" light and creates depth. The next mesh is the eye pupil - a simple circle. The pupil size is the same size as the iris hole. You can position it close to the inner side of the iris. And finally - the eyeball. It's a simple sphere with a hole in it.

I won't go deep into modelling of each element - it uses Blender's subdivision surfaces and it's quite simple as you can see.



The resulting hole will be ringed by 8 vertices (which you could subsequently extrude and use to help you model the sections below - just a suggestion for those who feel capable)

The **eye white** has white color and high values for Spec and Hard (depends on the lighting), as well as a moderately high Ref value (0.700 - 0.800). Optionally you can use a reflection map to make it look more wet but I usually don't do this.

(Noob note: Using a Phong specular shader turned to work quite well.)

(User: Make sure shadow buttons are off or it will black out the iris. Spec and Hard are located in the Shaders tab.)

2.66.2 Materials

Now let's look at the materials.

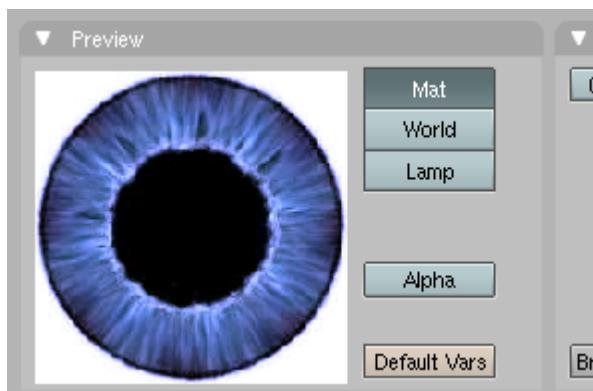
Iris

- Loop select the opening of the eye (ALT+RMB click edge of opening).
- Duplicate the loop (SHIFT+DKEY). Move the new loop away from the eye a bit perpendicular to the opening.
- With the loop still selected, extrude (EKEY). Without moving the extrusion, press ENTER.
- Resize by half (SKEY then .KEY then NUM5 then ENTER).

Sclera

To make the eye white, press NUM3 to go into side view and add a UVsphere with 8 segments and 8 rings. To create the hole at the front of the eyeball, delete the 8 triangular faces that make up one end of the eyeball. Ctrl+Tab+3 to go into face select mode, B for border mode and select the middle 8 triangles as shown below, press Delete or the X button and LMB on Faces.

- Move the inner loop back toward the eye a bit to form the concave shape.
- Create a new material and create a new texture called iris. Then hit F6 or click the spotty square icon  to open the texture panel, choose Image as the texture type and then press Load Image button and use the image below.



Here is the eye texture taken from the picture above



Alternatively, you can create an image like this procedurally, using the technique described [here](#).

All that is required now is a bit of tweaking of the texture size using the Xsize and Ysize values in the Map Input tab, and scaling the pupil hole size in Edit mode. Also please remember to change in the map input settings from 'flat' to 'cube'. You can tweak the RGB values and brightness/contrast of the image to achieve the appearance you want. Use a smaller value for Hard (about 50) otherwise you'll have a too shiny look instead of soft specular that fakes refracted light. The Spec value depends on the energy and distance of the light that illuminates it. Generally you'll need to take care that the refracted light on the iris should be no more than half as bright as the small specular spot on the cornea - otherwise you'll achieve the bad effect of two specular spots. Oh, another important thing - join the four meshes before tweaking the texture coordinates. Otherwise you'll have to do the job twice after you join them, because the texture space is changed. And activate shadeless button.

Pupil

The material for the pupil is a simple black color with the "shadeless" button on.

Cornea

The cornea uses a transparent material (**alpha = 0.1**) with **Spec = .6**, **Hard = 255** and **Spectra = 1**. "Ztransp" should also be turned on (found under Links and Pipelines tab). The cornea is simply a piece that fits exactly in the middle of the hole in the eye white. Make sure that the 'Traceable' button, under Render Pipeline, in the Links and Pipelines tab, is switched off. Noob note: I found it helps to turn off Shadebuff since, as far as I know, you don't generally want the cornea to cast a shadow.

Noob question: I don't know where is the "SpecTrance" Value. I'm working on 2.46. Thanks a lot. You can find "SpecTrance" under "Mirror Transp"

Noob Question: My cornea isn't transparent. I don't know where Traceable is since I'm using 2.6x

Noob answer: This applies to 2.6. Something that doesn't seem to be mentioned in this tutorial (sorry if it is and I missed it) is the cornea's shadow settings. If it casts a shadow, it will block out everything inside the eye. So under the cornea's Material->Shadow settings uncheck the cast box.

Another option is to leave the cast box in your cornea on, and change to shadow setting in your iris. Check the Receive Transparent, and that works just as well.

2.66.3 Lighting

The lighting is simple - move the eye to a new layer, create a new lamp and make the lamp affect only this layer. Position the lamp at a good angle so you have a small shiny spot of specular light on one side of the iris and a soft spot of "refracted" light on the other side. You can use a backlight to prevent the eyeball from being too dark at the non-illuminated part.

To move the newly created lamp to a new layer, press SHIFT+M and select the second blue button and press OK. P

Noob Question: Why do you need to create a new layer?

Noob Answer: It's just meant to prevent cluttering in your main workspace. It's completely optional.

That's it! Now you're (almost) ready to start with character animation. You have a nice eye, now you only need a character for it!

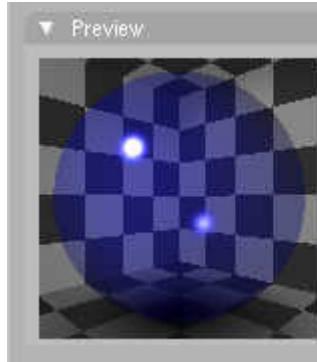
2.66.4 Changing the Eye Color

You can change the color of the eye by either changing the cornea color, or by changing the iris color itself. Changing the cornea color might be like putting on colored contacts.

Changing the Cornea Color

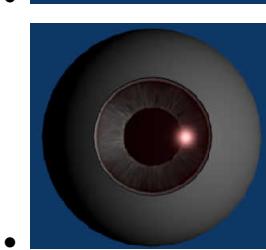
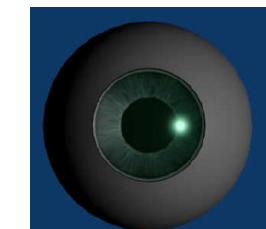
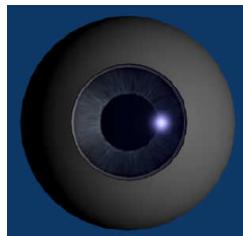
To change the cornea color, do the following:

1. In the mesh which is used to create Cornea set the value of the alpha slider to 0.2 (or more if you desire)
2. Change the color of the mesh to whatever color you like
3. Turn off traceable



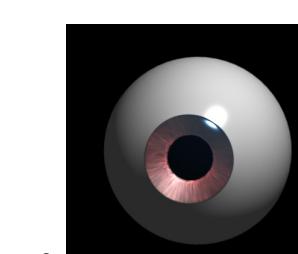
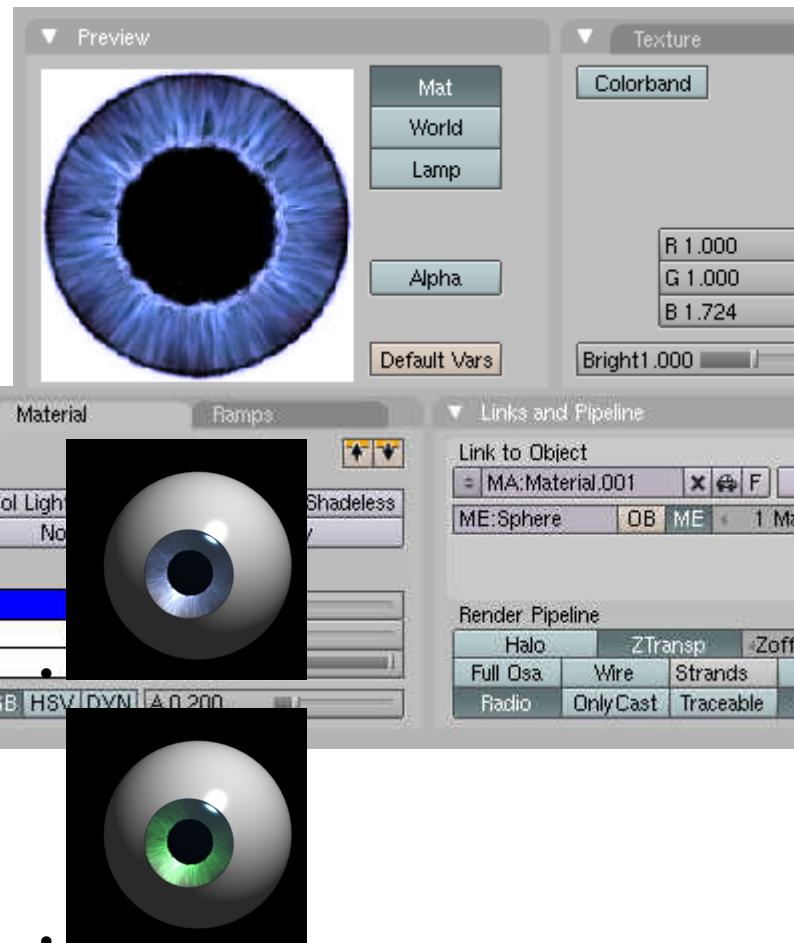
- 4. Render!

(and remember to smooth out everything by selecting the particular mesh going to edit buttons and clicking on set smooth)



Changing the Iris Color

To change the iris color you can either edit the iris image external from blender or you can modify the RGB values of the image when imported into Blender.



2.66.5 Pixar Eye Video Tutorial

- Ira Krakow's Blender 2.49 Pixar Eye Tutorial

2.66.6 Useful Links

For rigging an eyeball, (making it stay in one place while turning to look at something) and other eyeball stuff, you can visit this site. This tutorial assumes you already have the eyeball made above or one of your own. [Click. 5jun2009](#), the PDF shows no rigging, only creating an eyeball. (May 10th 2011, the site no longer exists.)

2.67 Procedural Eyeball

Level: intermediate

Building a better (procedural) eyeball!

Originally created by Jon McKay (Ammusionist) and posted on blenderartists.org/forum

The end result of this tutorial is an eyeball that fulfills the following requirements:

1. Single mesh

- I wanted to be able to append a single object into any project.

2. Procedural Textures

- I didn't want to have to rely on image maps that could get lost.

3. Versatile

- One single model to be used for any type of character, be it human, alien or whatever.

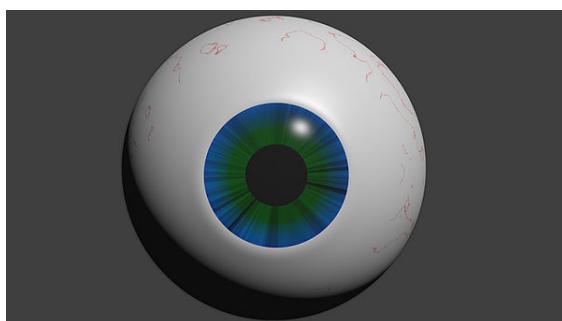
4. Easy

- Any effects needed to be quickly accessible. I don't want to be wasting time fiddling around with colour-band settings at the texture level

5. Impressive

- This sucker needs to look good any way it goes.

6. One other thing I really wanted was the iris musculature to follow the pupil dilation!



An old and different Tutorial:

1. Very old video for making deep-looking eyes []

2.67.1 Creating the Mesh

The objective here is to make a great looking eyeball. That means it may not be anatomically perfect, but there are some things we need to take into account: The eye

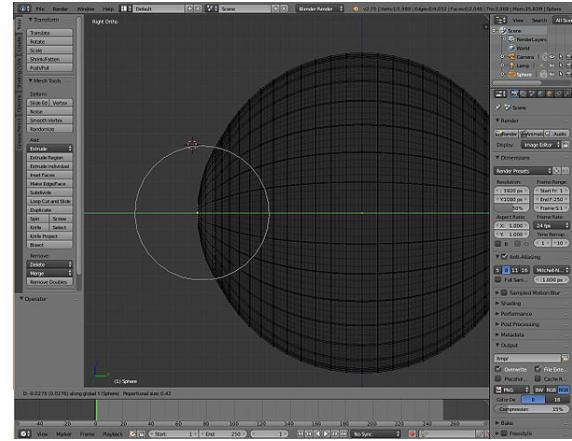
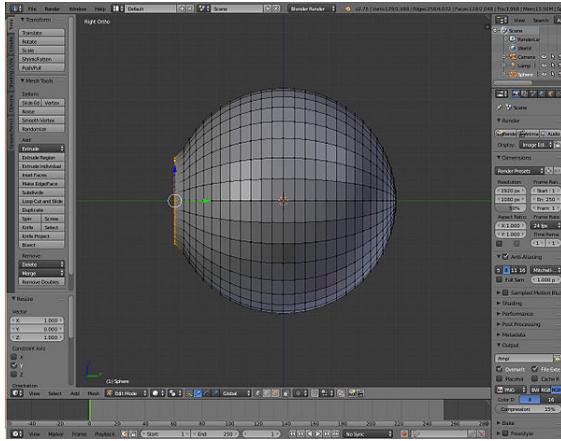
should consist of a white (white bit with veins); an iris (coloured bit); a pupil (black dot in the centre of the iris) and a cornea (clear bit surrounding the whole eye). These may not be absolutely correct, but these are how I'm referring to the pieces of the eye.

1. Start with a ball: Go to a front view (1KEY) and create a UV sphere. 32 Rings and 32 segments should be sufficient. It'll give us plenty of smoothness and enough verts to work with when we create shapes.
2. Click on "Align to view"
3. This sphere will become the coloured parts of the eye, but we also need a second sphere for the cornea. Use the Z key to switch to a wireframe view (if you're not already there). Hit the A key a couple of times to ensure all points are selected then Shift + D to duplicate the sphere.
4. Press S key to scale and scale the second sphere to slightly larger than the first. (Hint: Hold Shift while scaling for fine adjustment) or you could also enter a value, 1.01.
5. Now that we have a nice confusing mesh, we're going to make our life easier. We're going to hide the outer bit for the moment. Hit the H key.
6. **Noob Note:** Don't panic like I did the first time I accidentally hit H , you can bring it all back with Alt + H . (Note: In blender Alt + some key often reverses the effect of pressing the key in the first place!)
- OK Now to create the basic mesh for the inner part. Basically, we need to make a concave iris/pupil section.

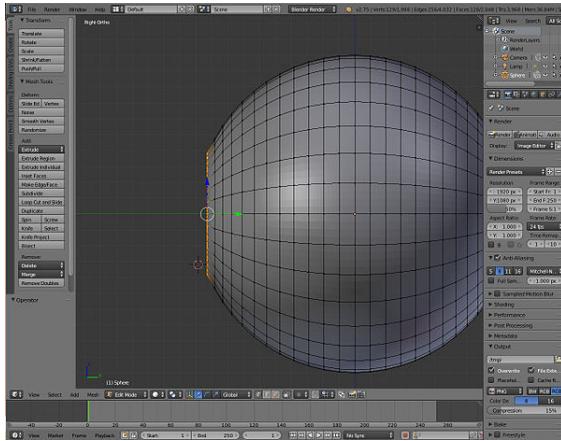
1. Make sure you're in Orthographic view.
2. Switch to a side view and use the Border Select (B) to select the first 4 rings and the tip. Now we're going to flatten these. Press S for scale Y for Y axis only and 0KEY for scale-to zero.

For more realistic eyes you can also grab the inner 3 rings + the tip a bit inwards the eye.

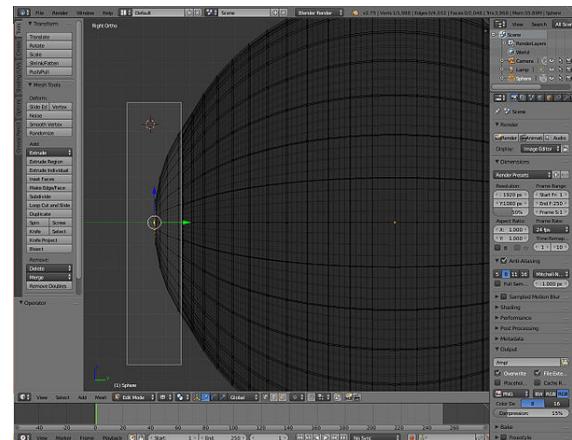
Noob Note: It is also possible to flatten the rings in a way that leaves the last ring in place and moving the rings later is not necessary. Before you flatten those 4 rings and the tip of the sphere place your 3D cursor in the plane where the last ring involved in scaling operation lies (use border select to select *only* the fourth ring, press Shift + S , and select "Cursor to Selected"). Then set the pivot point to "3D Cursor". Finally proceed like instructed above by selecting all rings to be flattened and scaling them to 0 in Y direction. This time the outermost ring will stay in place. Do not forget to set the pivot point back to the default option "Median Point". You may return the 3D cursor to the center by pressing Shift + C .



- Last, grab them along the Y-axis to the eye till it's round again.



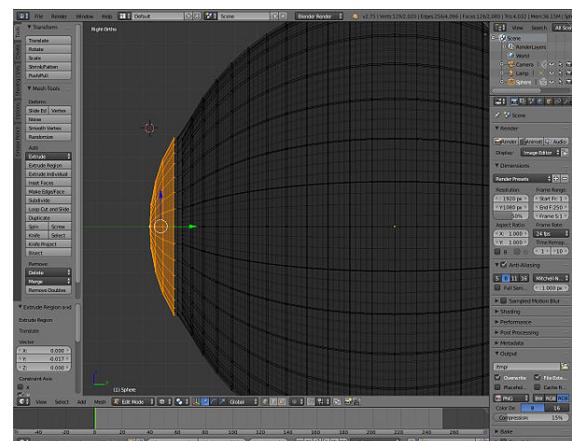
- First, switch off Proportional editing.
- Select from center the tip and the first 4 Loops.



- Now for the cornea. It's not perfectly spherical; it bulges slightly at the front. We're going to use the Proportional Editing tool.

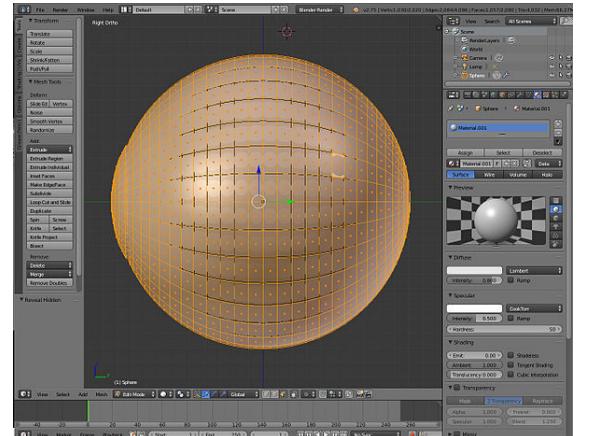
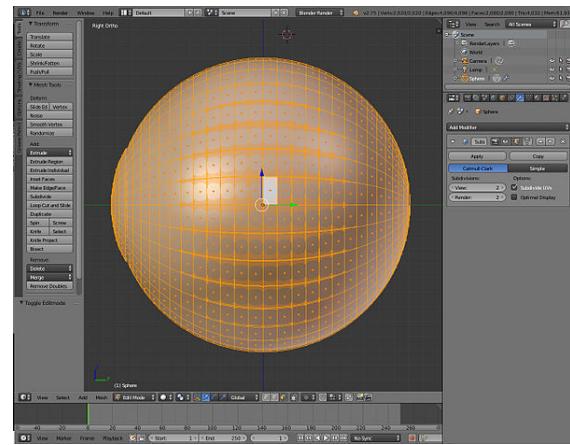
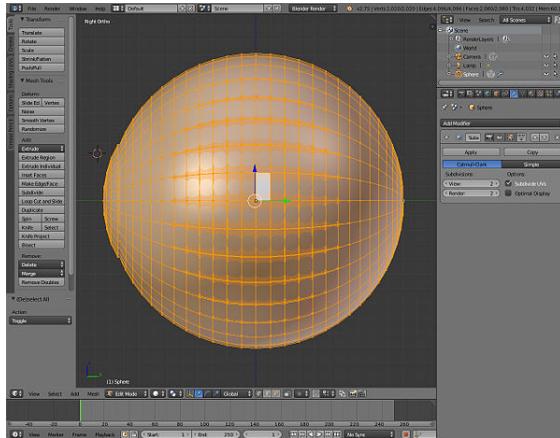
- First, we need to un-hide the cornea. Hit Alt-H to make it visible.
- Click the O-Key, this will activate proportional editing, click on "Connected" in the 3D view header at the settings for Proportional editing.
- Select the type of falloff: "Sphere Falloff"
- Press A-Key a couple of times again to make sure no points are selected, switch to a side view (Keypad 3) and right click just to the left of the front-most point. We just want to select the tip vertex.
- Now comes some fun. We want to drag that point away from the eye in the y axis, so press "G" for grab and "Y" for Y axis. As you move the vertex, you'll see a circle that defines the influence of the proportional editing. Adjust it so it is as the height of the whole iris and drag it out until it looks right.

- Then Hit Ekey and Ykey then drag out a bit 'till it's good.



- To clean the model up, we're going to apply a subdivision surface modifier and some smoothing.

1. Go to Modifiers in the Properties Header and hit new and choose “Subdivision Surface” choose view: 2.
2. Hit Tab to switch to Object mode and hit the “Set Smooth” button in the Link and Materials panel.



- Now would be a good time to save your work before we go on to textures.

2.67.2 Adding the Textures

You're going to need multiple materials on this mesh. It's possible to apply materials to selected faces.

1. Press A a couple of times to ensure nothing is selected.
2. Press Z to switch off wireframe; this will be easier without it.
3. Place the mouse pointer in the centre of one of the faces on the outside layer of the eye and press “L” to select linked faces. If you get a message saying “Nothing Indicated” try positioning the pointer on a different face and try again.

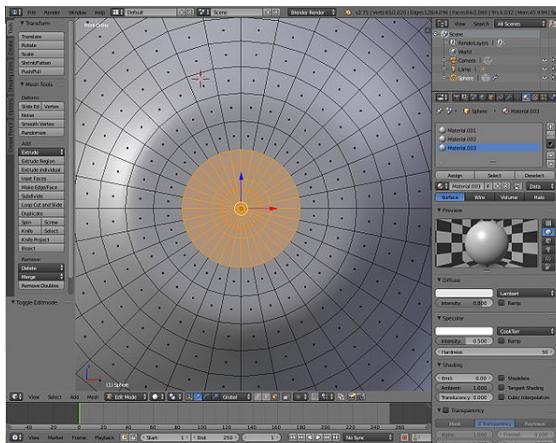
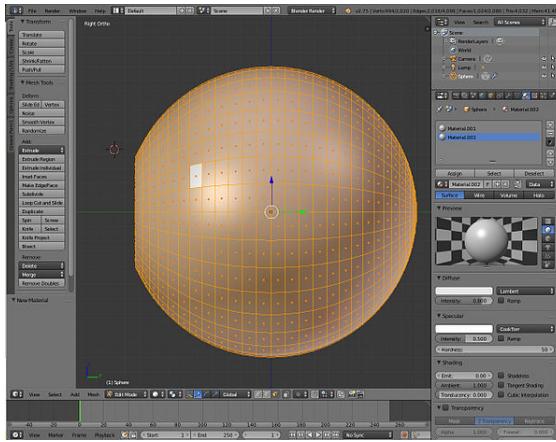
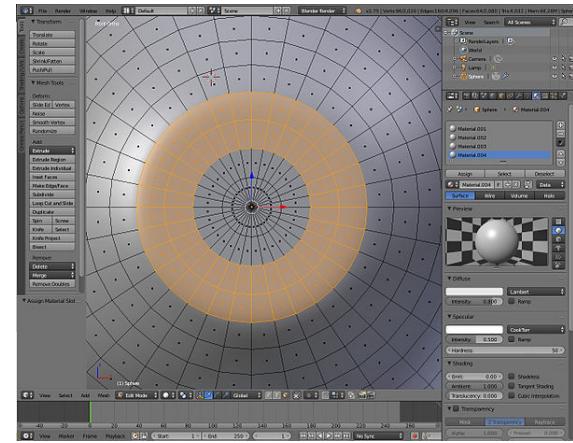
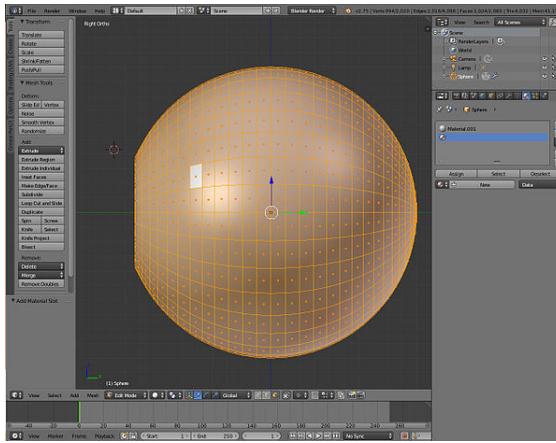
- Now the inside.

1. Hide the outside faces (H-Key). We'll assign the rest from the eye. If a face is already assigned to a material and you assign it to another, it just switches material.

1. Press A to select all and, as above, click “+” then click “Assign” so it is assigned to Material.002.

- Now for the iris and pupil.

1. Hit A to deselect all.
2. Switch to a front view and zoom in. Hit “C” to enable circle select and select the central faces plus one ring out. You can control the size of the selector by rotating your mouse wheel.



3. Again, create a “+” material index and “Assign” the pupil faces to it.

1. Finally, click Akey to deselect all and use circle select again to select the remaining faces on the iris. Assign them to a new material.

- Now we're ready to make the materials.

Cornea Material

1. Switch to object mode (Tab). Go to Material.001.

We'll start with the simplest texture first. The cornea is basically just colourless and transparent. This is because some lighting conditions make the eye under the cornea difficult to see. Go to the shading buttons (F5)

1. Locate the “Transparency” control. Set this on. This means as we go we'll be able to see through the cornea. Note that we're using Z-Transparency instead of Raytrace. And set Alpha to ".2"
2. Diffuse at white and intensity is 0.1.
3. Under Specular CookTor is used 1 and Hardness is 333.
4. Under Mirror, Click on, and set to 0.4, Depth is 3
5. Unclick the 'Traceable' setting under Options
6. And Under Shadow unclick “Cast”





White Material

The white of the eye's a little trickier. The material here includes some red veins that can be seen at the side of the eye, but not from the front. This can be done with two textures, but there's a cheat we can use.

First, the veins:

Switch to Material.002.

Go to the texture Properties aside of the from the Material:

Make a new texture.

Use a marble texture, and copy these settings.

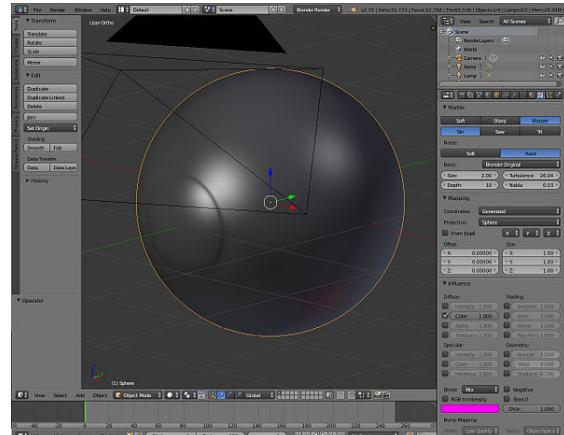
1. Under color choose Ramp.
2. The first point is "0": Pos=0; R=1, G=0.0, B=0.0; A=1;
3. Click on the second point, The second point is "1": Pos=0.340; R=1, G=1, B=1; A=1;
4. Under the Marble tab: Sharper, Sin, and under "noise", hard are clicked;
5. Size=2; NoiseDepth=10; Turbulence=26.04;
6. And set under mapping "Coordinates" to "Generated". And "projection" to "sphere" or "Flat".

The two colour band items are shown here separately. This makes a nice marble texture that will evenly cover the eyeball. So how to hide it from the front of the ball? Another colour band trick, but this time in the material itself.

Here they are in all their glory.

1. Under the material tab, Diffuse is white and intensity is 1;
2. Check Ramp, 0: Pos=0.533; R=1, G=1, B=1; A=0; Input is Shader;

1. 1: Pos=0.544; R=1, G=1, B=1; A=1; Input is Normal;



The colour band items are both white with one alpha 0.0 and one alpha 1.0. As you can see they are very close together to give a fairly clear line that the veins will stop at. The input setting for the band is set to "Normal" this means the left hand side of the band refers to faces that are parallel to the camera view and the right is faces that are facing directly towards the camera.

There's also some colour band fun to be had with the pupil material.

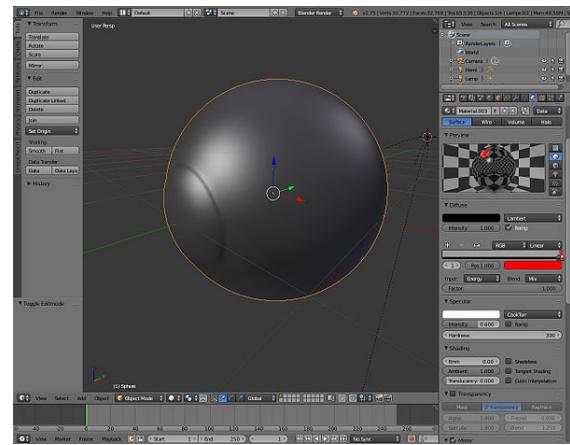
Pupil Material

I wanted a nice black texture, but I also wanted to get a red-eye effect if a light is shone directly at it from behind the camera. Why? I guess just because I can, and now so can you!

As before move to Material.003.

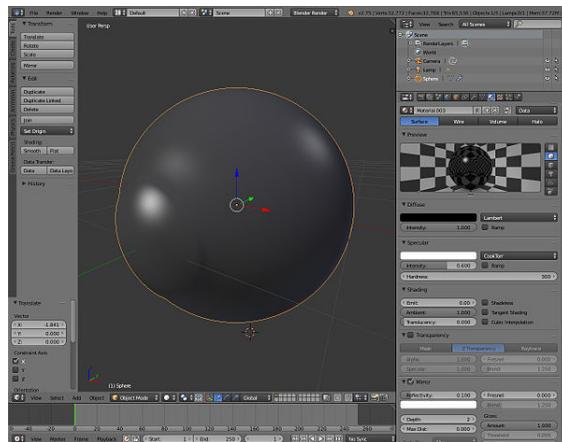
There are no textures in this one. It's basically black and a little reflective with a nice hard specular.

1. Diffuse is Black, presumably with an Intensity of 1
2. Specular is white with an Intensity of 0.6 and a Hardness of 300;
3. Under Mirror reflectivity is 0.1;



We need the texture to stay glued to the mesh irrespective of the shape of the mesh. This gives the effect of the muscles in the iris expanding and contracting. In a nutshell, we're going to UV map a procedural texture to a single part of the mesh.

OK, here we go. Switch material.004. You'll need to be able to see both 3D view and UV/Image editor. This can be achieved by splitting windows, if you don't know anymore how to do this read [the “Blender Windowing System” module](#) again. Noob note: If you rendered the image of the eye you will see the rendered image, we want an empty grid. So click the “X” in the Datablock (Image) choose-section in the UV/Image editor Header.



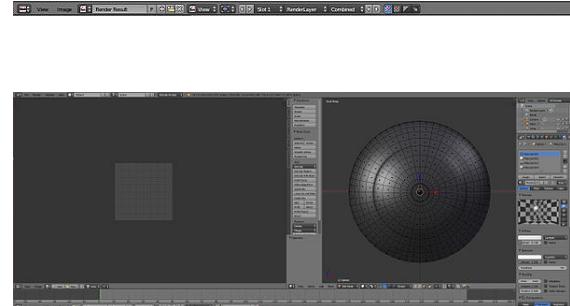
If you want to have the red-eye effect, have a good look at the colour band settings though! (Not Necessary, and maybe not wanted by everyone)

1. Check Ramp.
2. 0: Pos=0.970; R=0, G=0, B=0, A=0;
3. The second is fully red (with no blue or green). We want the red-eye to respond to light not angle like the white of the eye.
4. 1: Pos=1; R=1, G=0, B=0, A=1; Input: Energy;

Now the bit you've all been waiting for – the iris!

Iris UV Map

Save your work and grab a cup of coffee for this part. It's a little involved.



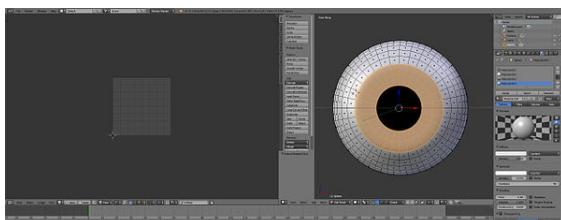
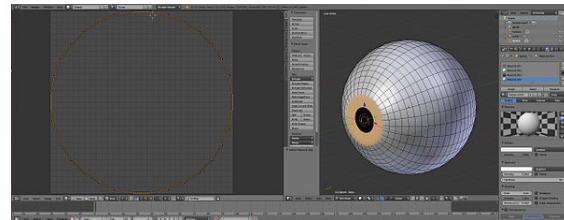
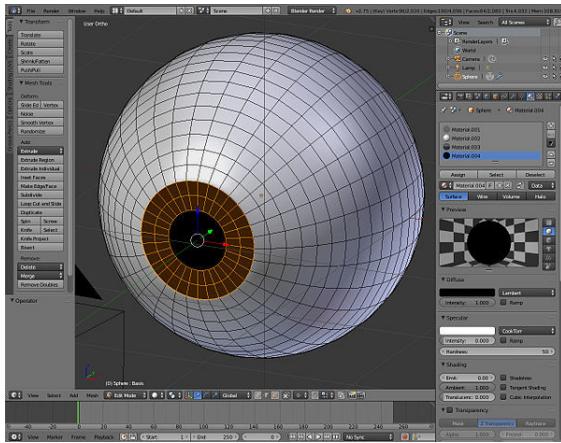
Make sure the window is displaying a front view (Keypad-1) this will be important when we unwrap the bit we're going to use.

Change the 3D view to edit mode en Face select mode.

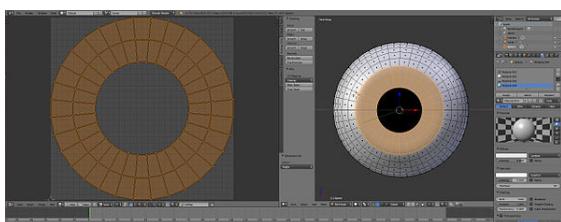
1. Hide the outside (cornea) as done before.
2. Go to Material.004 and click on “Select” right of “Assign”, this will select the to the material assigned faces. We will unwrap the faces to place our texture.

- You can use these properties for the iris material:

1. Diffuse: black but with an intensity of 1
2. Specular: Intensity: 0



1. Now search in the tool shelf for the Shading/UV's tab, you will find there the "Unwrap" button under "UVs".
2. Click it and select "Unwrap". The unwrap tool, is a tool which you can position a texture. You can allocate the places in the texture to the mesh by positioning the unwrapped faces in a grid. The texture can be an image or another texture and will be used at the places, that are described in the unwrapped image in the "UV/Image editor".
3. Zoom in 'till it's placed exactly to the outer edge the "UV/image editor" window.



1. The image window should now contain a nice even circle of dots. If they aren't selected anymore, put the cursor in the "UV/Image editor" and press "A" to select all the points. Now select each ring in turn using Alt-Shift, starting from the innermost, and scale them until they cover from the centre to the outside of the image grid as shown. (This won't affect the mesh)

Now we've got a map, lets get a texture. In the 3D view "Tab" to object mode for the next part. If the UV/Image editor is in your way you can merge the two windows. If you don't know anymore how to do this read the "Blender Windowing System" module again.

Iris Texture

Firstly, a texture to control the colour blend across the iris. Generally it starts on the outer edge dark and gets lighter as it moves to the centre. Sounds like a job for the blend texture.

1. Create a new texture for Material.004 at the Texture Properties in the Properties Header. It will be called "Texture.001".
2. Change Type to Blend.
3. create a second texture with type: Clouds

The texture itself is fairly straight forward, but again, the colour band is where the magic happens.

There are two stops in the colour band. Note the alpha values. This blend texture is meant to tint the iris texture so it's lighter in the centre and darker in the outside.

We make a Blend texture and a clouds texture (yes clouds)

Now for that iris texture:

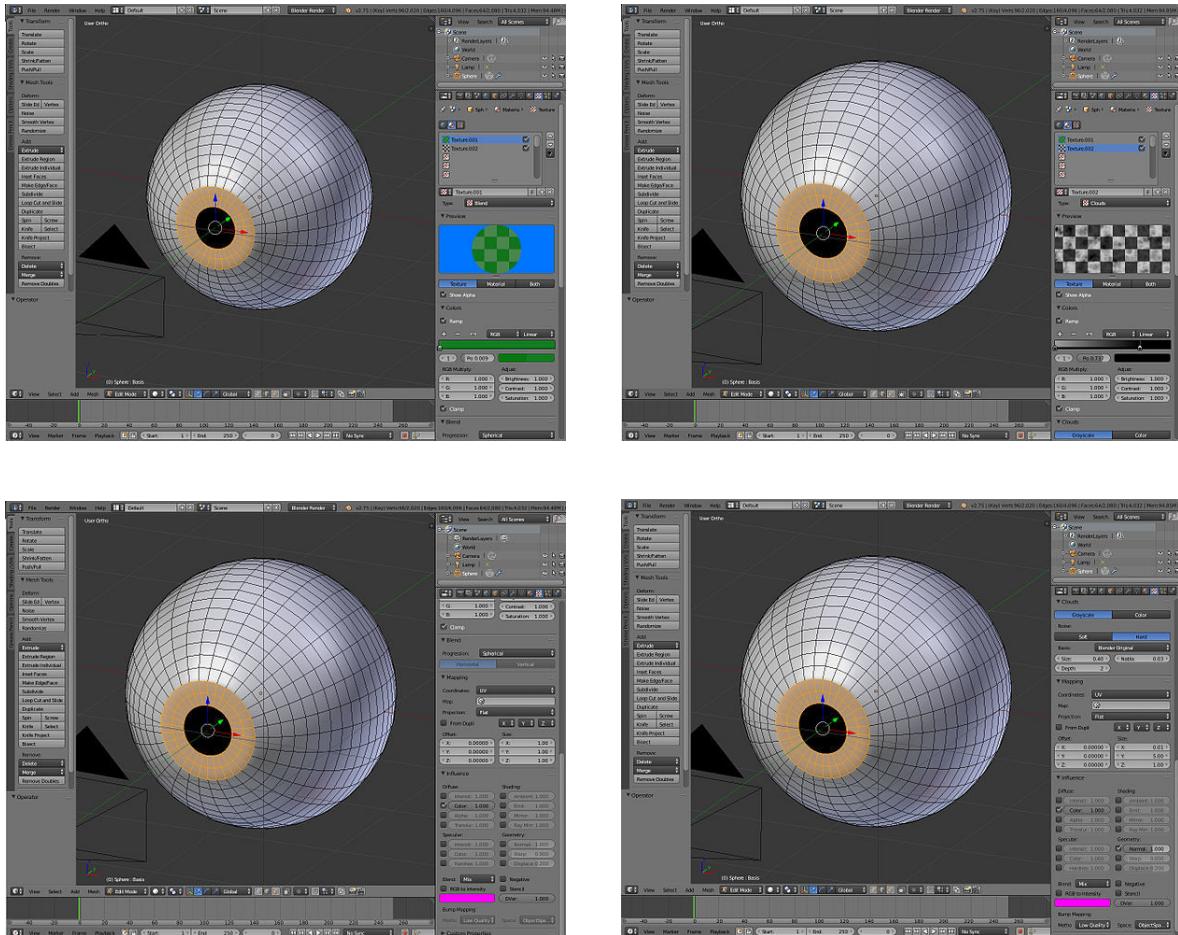
Blend texture:

The blend texture is applied to the UV map input. The green colour is used as a lighten mix to tint the iris near the centre.

1. Select Ramp,
2. (Blue) 0: pos: 0, play with the color settings and set alpha to 1
3. (Green) 1: pos: 0.009, play with the color settings and set alpha to 0.89
4. set progression to Spherical
5. Set mapping to Coordinates: UV, Projection: Flat

Now you've got color

Clouds texture:



The colour band points are black with alpha 0 and white with alpha 1.

1. 0: pos: 0, black with alpha: 0
2. 1: pos: 0.737, black with alpha: 1
3. Under Clouds: Grayscale, Noise: Hard
4. size: 0.4, depth: 2
5. here is (together with the UV mapping) the magic of the lines: Mapping Coordinates: UV, Projection: Flat
6. **Important:** Size X: 0.01, Y: 5
7. Under influence - Geometry - Normal: 1

This texture is applied to UV again and. This makes some small shadows in the iris when the light is from the side. Add some lights and have a look at your eyeball so far.

2.67.3 Pupil Dilation

We're going to use shape keys to create the pupil effects.

We're going to create a normal eye and a cat's eye. Fortunately, the fully dilated eye is the same in both cases so that's going to be our shape key basis.

Noob Note: note that you can only set a "Shape key" in object mode.

1. Go to Object Mode and go to the "Shape Keys" properties in the "Object data" (The triangle one) context in the "properties" header.
2. Click "Add Shape Key" to create the "basis".
3. Make 3 extra Shape keys. (Don't make a shape key after editing, because you will change the previous keyframe and that is not the intention)

1. Go to "edit mode" and click on Shape Key 1.
2. Use circle select, Alt+Shift of de Select button in Material to select just the first 2 loops en the tip (pupil) in vertex select mode.
3. Now scale this to the next loop, select the latter and scale to just before the edge of the iris (This gives a very wide pupil). It has to be as wide as possible because you can adjust how wide you want it in the

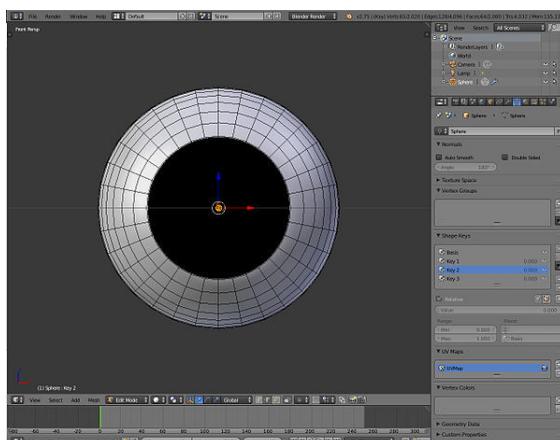


shape key section. It is just a maximum so you can simply make an animation.

4. Change the name to “Wide Pupil Contract”



1. Just because you may also need it make a very little pupil (with shape key 2). Call it (Little Pupil).

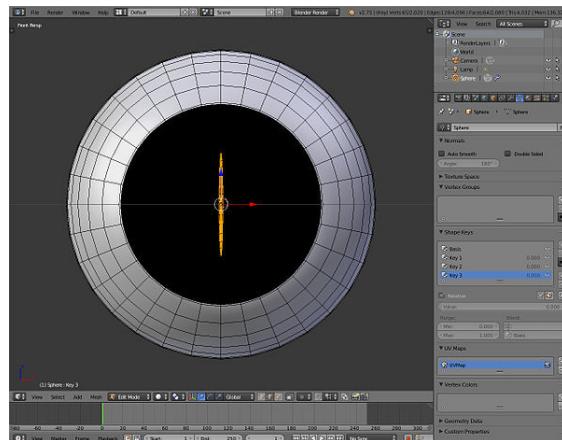


Use the value slider in “Object mode”, also set Viewport shading to Rendered (so you can see how it will be when

rendered), To increase the shape of the pupil with “Wide pupil Contract” and make it smaller with “Little pupil” you can use them at the same time also (and much more if preferred).

Now switch back to object mode and try the shape key sliders out. Pretty neat huh?

1. Let's make a cat's eye now. Change the name of Shape Key 3 to -“Cat Pupil”.
2. Use the same method as before to scale the edge loops. As you scale, restrict your scaling to the X axis (That is “S” then “X”). You should end up with a shape key tiny but medium long.

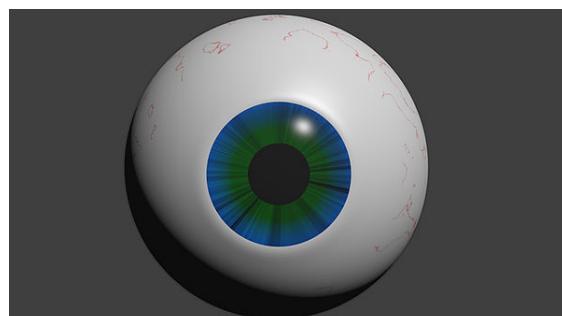


That's pretty much it. Try some other things to practise with this.

Set the Camera and lightning.

Note: Due to the settings you've done in the material of the white of the eye, you have to position the camera in front of the eye. Because the white of the eye that overrules the vein-texture is a white spot in the middle of the camera and that has to be placed in the middle of the eye around the iris.

Don't forget to save – and enjoy!



2.68 Putting It All Together: A Dragon!

2.69 Putting it all Together: A Dragon!

I downloaded Blender without knowing much about how to use it, probably like you. I hunted around for a good tutorial and found this one. It is teaching me wonderful things! I have diligently followed every lesson from the beginning, as I hope you have. Every lesson seems to contain some valuable technique. So now it is time to put it all together into a single project.

Hopefully, you now know

- How to add objects, move and rotate them, scale them in various dimensions, and change their shape.
- How to use proportional editing to change them even more.
- How to parent, join and separate objects.
- How to use materials and textures.
- How to use images in materials.
- How to use basic lights.
- How to use modifiers including subsurf and array.
- Basic use of curves
- It will help if you know how to use a photo editor such as GIMP to produce seamless textures

This project will build on all that background. Since all those things have been covered before, this will not be a key-by-key tutorial but rather a description of the general steps to take to make the finished scene.

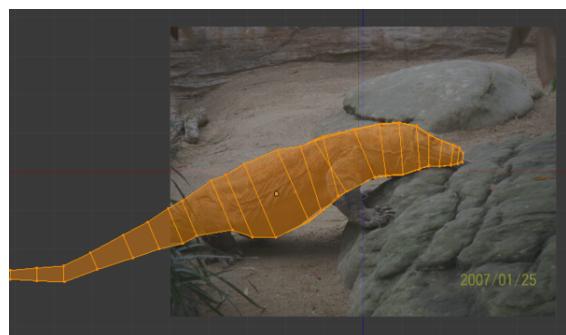
2.69.1 Modeling from the Image

It happens that the Komodo Dragon is one of my favorite creatures. Let us start with that and let our imagination run wild. In the lesson on **Modeling a Fox from Guide**

Images we learned how to make a beautiful 3 dimensional wolf from an image. (Yes, I know. The fox turned into a wolf. If you did that lesson you understand.) Here is a nice picture of a Komodo Dragon by wallygrom from **Creative Commons** with a proper license. We can then set that picture as a background image and apply the technique of starting with a cube and scaling it, extruding it, scaling again and so on. We only have one view, so our imagination and application of other pictures has to help with the top and end views, and we have to add a nice long tail. As you build your cubes keep in mind that you



Komodo Dragon



Using cubes to model the dragon



The basic body smoothed.

will have to create faces and extrude the legs so place a well sized face at the proper spot.

We will use the techniques we learned to split faces and add the legs. Be sure to provide a nice mouth to put teeth in later, and pay particular attention to modeling the claws. I modeled one entire foot and then duplicated it three times to have them all, and joined each one to the body at the appropriate spot.

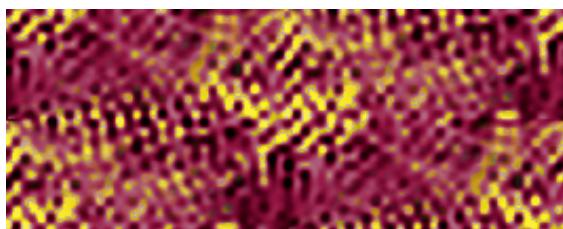
Once you have outlined the dragon with the cubes, remember what we learned in **Smoothing Your Simple Person** and apply subsurf modifier and smooth the entire thing. You can then push and pull on vertices to help

make it look as good as possible. I wanted to try to maintain the look of an ordinary creature and stay close to my Komodo Dragon, rather than a huge fierce beast to take on St. George.

2.69.2 Setting the Body Materials and Textures

Use what we learned in **Image Textures** to give the dragon his scales. The most used technique in this tutorial is:

- Search **Flicker Creative Commons** for a properly licensed picture that has a good material in it. If you use Google or some other search, be sure that the picture you get is properly licensed if you plan to share the image.
- If your image is NOT a single seamless texture, use photo editing software (such as GIMP) to produce a usable seamless file.
 - Load the image into GIMP. Crop it to a reasonable size showing a relatively uniform texture.
 - Choose *Filters > Map > Make Seamless*. Other photo editors will usually have a similar operation.
 - Save the image with a suitable file name.
- Add a material and a texture using the image. Adjust the settings as needed to look good.
- If you are following my steps, I have put all the textures in this article for you to copy if you wish. Click on the thumbnail image and go to the Wiki page from which you can download.



Fantasy Dragon Skin seamless texture

You can just select a rectangular swatch of skin from the picture and use GIMP to make it seamless. But Komodo Dragons are a depressing color. For a fantasy creature we need a brighter skin! So you can play with GIMP Colorizer until you have a suitably colored swatch. I used red and gold, but use your imagination! This becomes the first material and texture for the dragon's body. It will probably be necessary to repeat the image several times in both the X and Y directions. I used X:6 and Y:2 in this particular case.

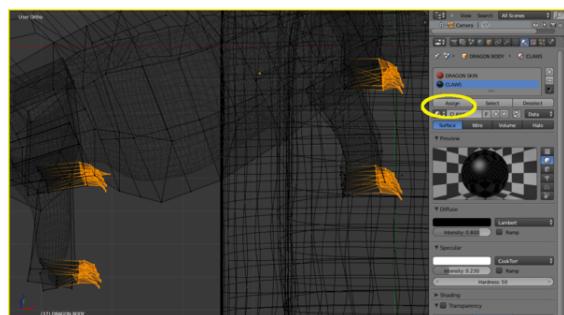
2.69.3 Adding the Eyes



The first rendering.

The dragon needs suitably scary eyes. Hopefully you have already modeled eyes for the **Procedural Eyeball** tutorial. I chose the cat's eye. In **Some Assembly Required** we learned how to append a file. Append the Cat's Eye, and then work on the color textures until you have a suitably scary dragon eye. Then duplicate it and put one eye on each side of the dragon's head. I joined the eyes to the body, but you can also just parent them if you are careful. You will notice in later renderings that I kept playing with the eye color and position.

2.69.4 A Dragon Needs Teeth and Claws



Assigning texture to the claws



Four rows of Teeth

Using the technique from **Multiple Materials per Object**, isolate the claws and give them a nice shiny black

material. Too much Specularity will make the claws look like plastic, too little will make them dull and uninteresting. I added just a little bit of Mirror also. It will be something to experiment with until you get the look you like. Click on the picture to enlarge it and see the setting I settled on. Remember that you have to select the area of the mesh and ASSIGN it to the material or you will wind up with a dragon that is black all over! (That is the voice of sad experience speaking).

A dragon needs teeth. A lot of teeth! Modeling all those teeth would be a big job. Fortunately we learned how to use array modifiers in **Building a House**. Add a cone, scale it down very small, and extrude some more parts so you can shape it into a nice curve. When you have a beautifully sharp tooth, add the Array modifier and make it into a whole row of sharp teeth. Then duplicate the whole thing three times, rotate it around as needed, and you will have four rows of teeth, enough for both jaws. This is why we left the mouth open when we made the original model.

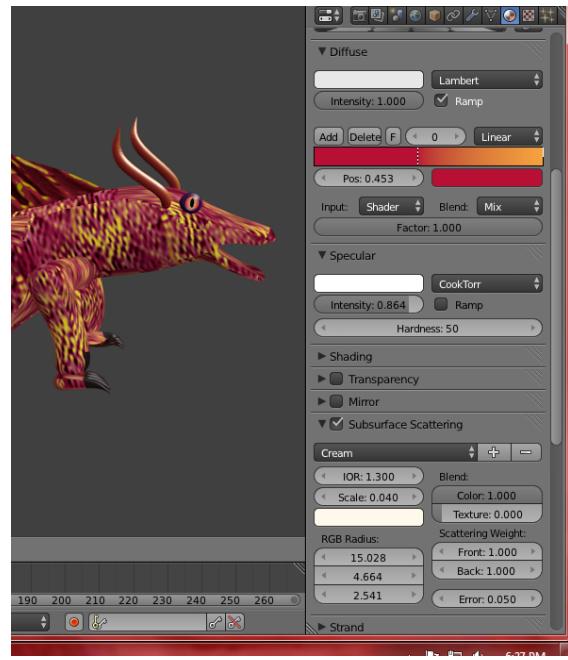
2.69.5 Horns, a Crest, and Nostrils



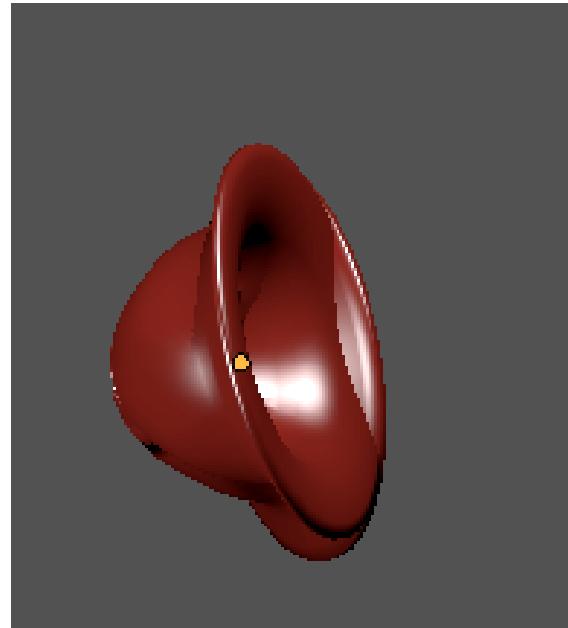
Using the Curve Modifier

Lets give the beast some horns. Here we can use what we learned in **Deforming Meshes using the Curve Modifier** to make some gracefully curved horns. Just as in that lesson, add a cone and extrude segments to give a lot of vertices. Then create your Bezier Curve and set it as a modifier to the horn mesh. Properly done your horn will mold itself to the shape of the curve. Duplicate it and rotate it to fit the other side of the head and put them in place.

Animal horns are some of the most beautiful structures in nature. We should make a particularly pretty material for them. We can use the blending colorband as in the tutorial **Basic Carpet Texture**, updating it a bit for 2.69, to give the horns a nice color appearance. Then add **Subsurface Scattering** as taught in that tutorial, with a lot of experimenting, until you get a really beautiful appearance. The settings I used are in the picture but it is fun to try different things.



The color settings for the horns

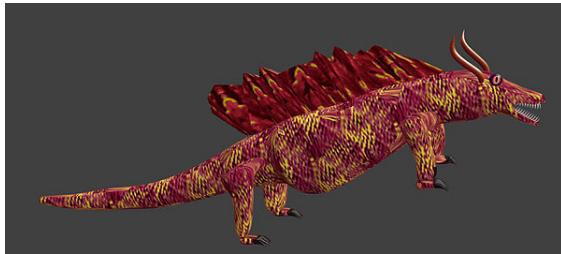


Nostrils

An impressive crest should dominate the dragon's back. I found a picture of a basilisk lizard and used it as a background model for the crest. The Crest texture was produced in GIMP exactly the same way as the dragon skin texture.

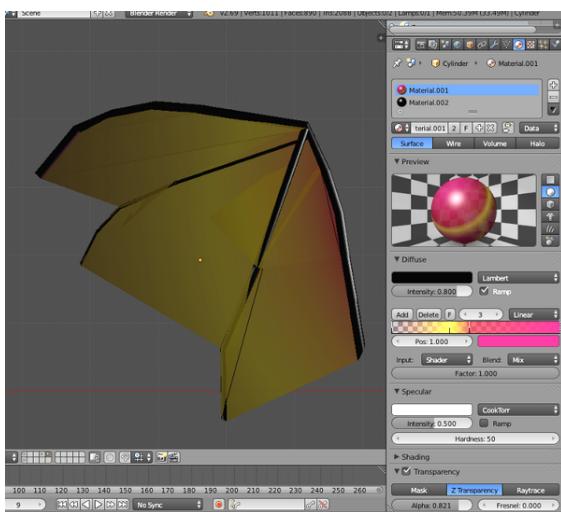
Reptile nostrils are sort of nondescript, so lets make some nice nostrils like a snorting bull might have. You can use the technique from **Spin a goblet** to model one nostril, then duplicate it, rotate the pair until they look fierce, and give them a nice color like the dark red brown I picked. Then put them in place and parent them to the body.

Now you can render your dragon and see the basic body shape, but something is missing. What is it?



Teeth, Claws, Crest and Horns

2.69.6 Giving the Dragon Wings



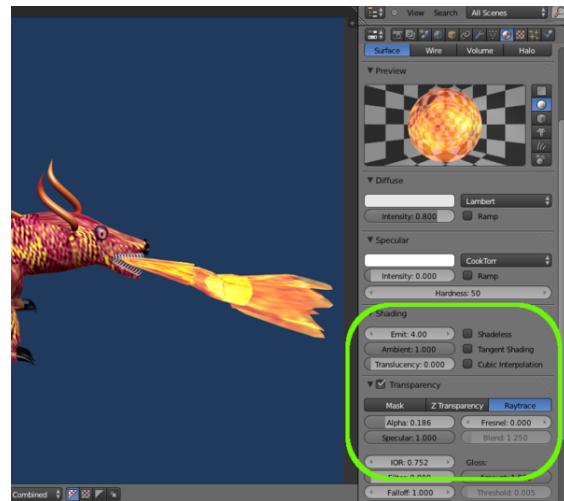
An attempt at wings.

The wings were the hardest part, because we have not had many tutorials in this series that dealt with the problems involved. I started with a cylinder object and went from there, joining other cylinders to it, and then selecting edges and using F to create faces. There may be other better ways to do it. I wanted the wing sail parts to be semi transparent and give an appearance of shimmering colors. Experimenting with the transparency and the materials as seen in the screenshot gave me some I liked pretty well. I am not completely satisfied with it but perhaps it is the best that can be done with the tools we have so far learned. Joining the wings to the body seemed to produce some distortions so I just parented both of them to the body object.

Now there is just one more thing that a perfect dragon needs.

2.69.7 FIRE!

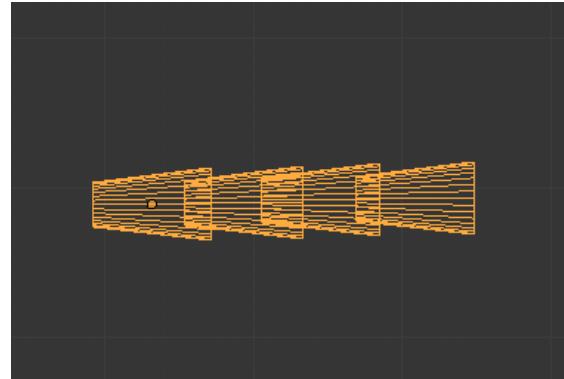
Everybody knows that dragons breathe fire. It seems that there are many different ways to do flames in Blender.



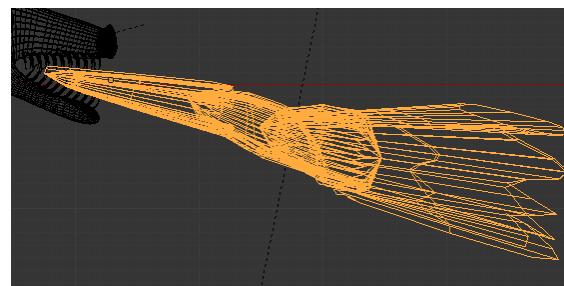
Settings for flames



Seamless flame material



Scaling the cylinders



Distorting the cylinders

However, I am attempting to use methods covered so far in this tutorial series, so I wanted to experiment and see

what could be done with just what we have learned. I came up with something reasonably close to flame breath using basic techniques. I started with a series of cylinder objects, scaled at one end until they are almost cones, and then stacking them inside each other. I used the proportional editing that we learned about in **Mountains Out Of Molehills** to pull and stretch into some more or less random shapes. I found the flame picture and turned it into a seamless material swatch. Then I wrapped it around my cones and began playing with the settings for transparency, and added an Emit setting just to see how it looked. (If you want to jump ahead refer to **Making Fire** in the next section of this series.)

2.69.8 Ground to Stand on



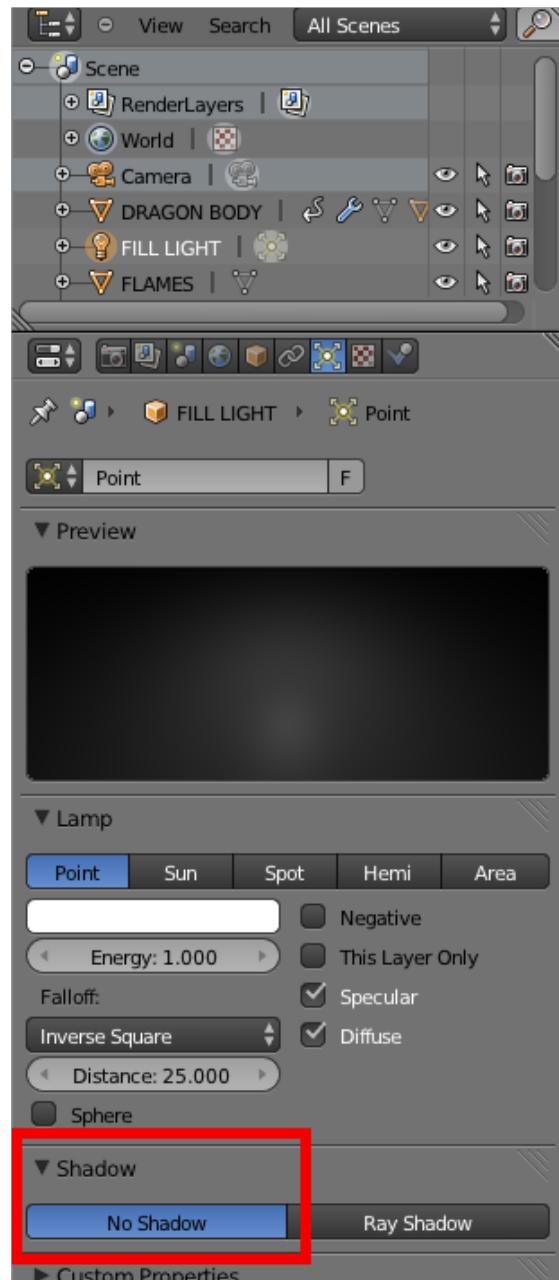
seamless lava texture



seamless grass texture

Now it is time to complete the scene by giving him something to stand on. Since his front feet are raised, put in a sort of lava flow for him. That seems appropriate for a dragon. Try Adding a series of cubes. Stack them up and pull them out of shape with proportional editing. Add a subsurf modifier, and then set the lava rock as an image texture. Again I made a seamless material out of a photo.

Then create a plane and again use the proportional editor to make some nice hills. A seamless green grass texture repeated multiple times completed the ground.



The No Shadow setting

2.69.9 Lighting

So far the lighting has just been the default point light. I think it is a bit more dramatic to use the Sun style of lamp. As we learned in **Light a Silver Goblet** go to **Add>Lamp>Sun** and place a sun type lamp rather high in the sky. You will notice that now you get two overlapping shadows, one from the Sun and one from the default point lamp. You could just delete the point lamp, but, to advance beyond what has already been covered, it is useful as fill lighting. You can select the point lamp, go to the properties panel and click on the light icon . Scroll down to **Shadow** and click **No Shadow** and the point light

becomes a fill light. Setting the horizon color in the world panel  is the last step to produce this scene.

2.70 THE FINAL RESULT



The Final Result

I think it nicely combines pretty much all I have learned about Blender so far in the tutorial! If you want to try your hand at a project, it is not as hard as you might think. You and I have really learned a lot by going through the first two sections of **Blender 3D:Noob to Pro**. I thank all the authors who have worked hard to bring us to this point.

I challenge you to show off what you can do with your newly acquired knowledge and put your finished work up here. It will probably be a lot better than mine.

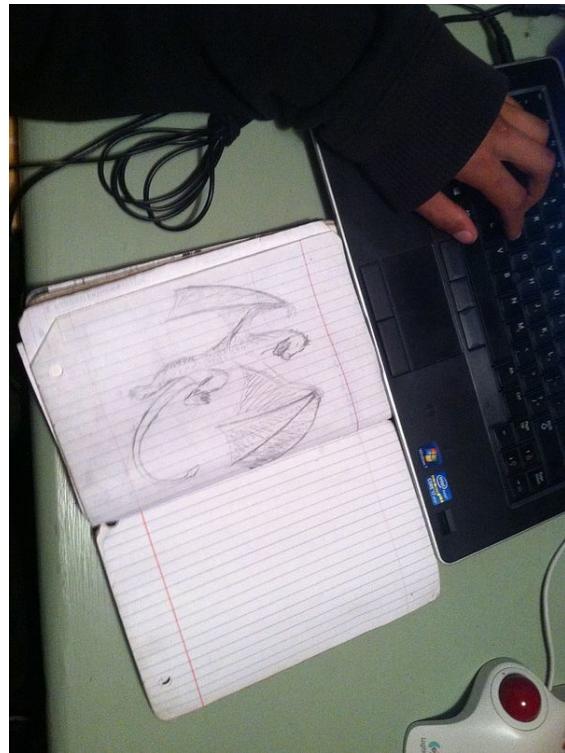
2.71 Noob Project No.02 - Space Dragon in Space

First off, thanks for creating this page! What a wonderful idea to include all the good readers in a final project! This is what openness is all about, and this wiki platform is the perfect place to be doing it. Also, huge thanks to whoever originally wrote this tutorial. Whoever you are, nameless stranger, you're doing excellent work.

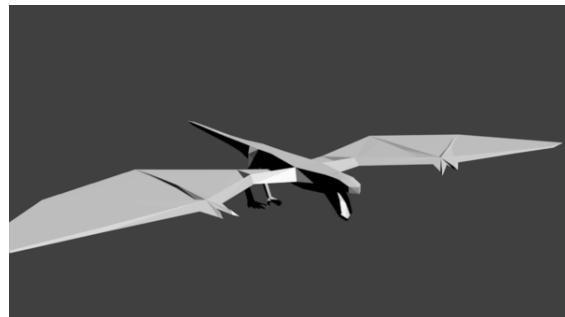
Moving on, I knew I wanted my dragon to live in space and shoot lasers, so the first thing I did was sketch him out so I had some sort of plan before beginning the tedious busy-work.

I think this is technically called a “wyvern.” Anyway, with this general idea in mind, I began extruding a cube to bound out the basic shape, keeping in mind how its internal skeleton would look and behave (if dragons were real of course) The wing hand and knuckle bits were kind of tricky. For those, I extruded a long rectangle for the arm, and added a few loop-cuts at the end, which I extruded into fingers. Tricky and finicky.

There's probably a better way to do this. While editing, I recommend using a mirror modifier, but be careful of



A Quick Sketch



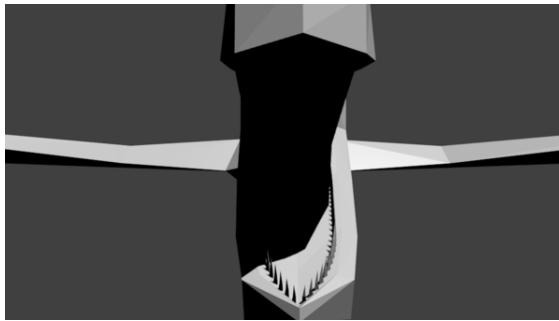
He looks like a chestburster with wings.

vertices on the negative side of your mirror. Best thing to do would be cut your model in half along its axis of symmetry and then apply the mirror. Otherwise, vertices can show up on the positive side where you don't notice them until you're almost done modeling. Have fun cleaning that up... I did. **Select > Select All by Trait >** helps a lot with this, and as always, remember to remove doubles every now and then! After several hours, here's the basic, blocky model of a wyvern I made.

Next I made some teeth. Cone with 3 verts > Curve modifier > Array x 37

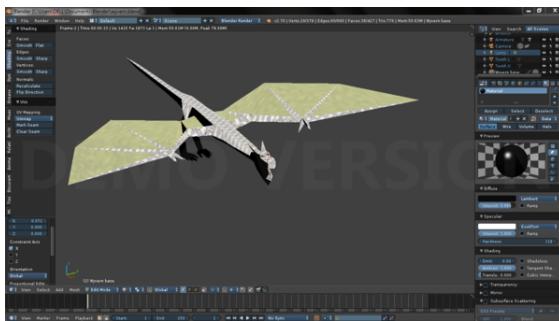
Now, those teeth look a little small, and I suppose 37 rows of teeth is a little much for anyone, even if their mouth is 4.76 meters long. Next time, I'll probably give him less teeth, and just make them all a little bigger.

I wanted to make a U/V map and draw some real nice textures on this thing, but we hadn't gotten there in the



What a pretty smile!

tutorial yet, and I figured I'd follow the precedent. Besides, I don't properly know how to do one anyway. So I opened up the GIMP and drew some neat looking scale textures. Actually, first, I looked up scales on wikipedia (and learned that scales grow from the epidermis and are weaker than *scutes*, which grow from the deeper dermis, and so are more durable, among other differences. It's cool stuff, for sure.) So I pulled up some images of scutes from the internet, and I got to work. I made my textures super low resolution (32x32 I think) cause I wanted that retro/late-90s rendering look. Haven't you heard? It's all the rage these days! It also helps with rendering times on my cryptop computer. Just check out this YouTube playlist I was listening to while I made it.



Now with textures and materials! ...and 90% less sugar

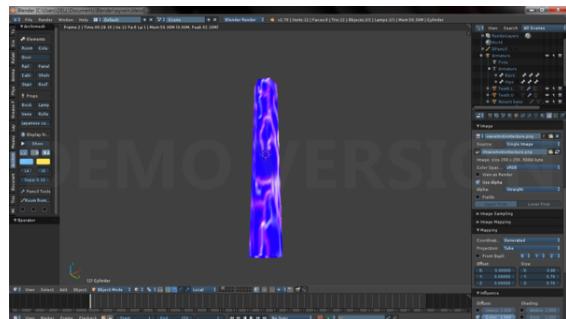
Of course that strange position he's in looks uncomfortable, not to mention it would probably make for a unexciting rendering. So I gave him some bones for him to orient into a more natural position with. The toughest thing about making an armature has got to be symmetry. At its very best, it's tedious. At its worst, it's torture. I know there's a way to edit armatures with a mirror modifier in blender, but I couldn't find it, and anyway, I finished up alright. A thing to note: If you've gotten pretty far along and you find you need to delete a bone in the middle of your armature, you can rejoin the ends by selecting them and hitting **Ctrl + P > Connected** To keep your armature in the same position, reposition the ends with cursor to selected then selected to cursor.

Next, I wanted him to shoot lasers out of his mouth instead of fire, cause ya know... He's in space. So I opened up the GIMP again and made a texture to wrap around a



Now he's got some bones on that meat!

tube that would be the beam.



That didn't come out right...

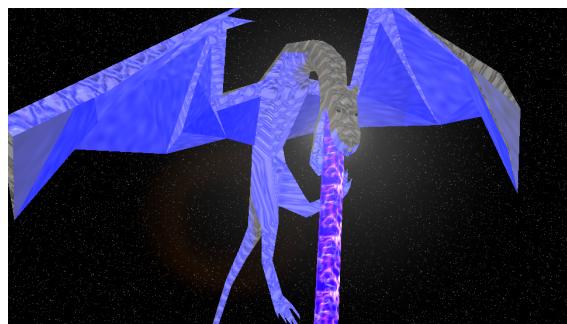
Lots of lightning and sparks everywhere... It was cool. The beam itself was a tube with a blue material on it -- full emit, can't receive shadows, slightly transparent, and full translucency. I slapped my texture on and put the tube in the beast's mouth. I duplicated that tube with **Alt + D** and shrunk it a little on its local !Z axis (**Shift + D**) so it would shine through. Inside the dragon's mouth and down the length of the tube i stuck a few point lights with the energy turned up to 11.

Actually it only goes to 10. Maybe those guys at blender should fix that.

To get the beautiful starfield you see in the background I went into the GIMP, filters > noise > hurl (what a great name for this thing btw), and desaturated the picture with colors > desaturate. Yep. Simple as that. There's probably something you can do right from Blender with procedural textures, but either way, sky texture! getting it to show up in the view took me some searching. I didn't remember how we'd done it from earlier in the tutorial. Anyway that sort of repetition helps the memory.

AAAAAAAAAHHHH!!!

Add your project below this line



AAAAAAHHHH!!

Chapter 3

Unit 3 - Broadening Horizons

3.1 The UV/Image Editor

Blender's UV/Image Editor window serves a number of related purposes:

- It is a place to view rendered images, and save them to files.
- It is a place to view and edit images being used as textures.
- It is a place to perform *UV mapping* of meshes to texture images.

In the window header, you will see "View" and "Image" menus, followed by a menu for choosing which image to view. The rest of the window header varies depending on your choice from this last menu.

3.1.1 Viewing Render Results



Here is what the variable part of the window header looks like when you choose the special "Render Result" entry in the image-selection menu. The next menu, showing the word "View", serves no useful purpose in this case (it is only applicable to editable images, not the render result). Next to that is a Slot menu, allowing you to select from any of 8 numbered slots. Each slot can hold a separate Render Result image. This is handy for trying out different render settings (e.g. quality etc), and quickly flip between them for comparison. You can also use the keyboard shortcuts 1KEY .. 8KEY to switch slots. When you perform a render, the generated image is put into the slot you are currently viewing in the UV/Image Editor.

The next two menus, showing "RenderLayer" and "Combined" in the above screenshot, allow you to view different render layers and render passes, if you have more than one of these configured in the render settings, as well as the final compositor result.

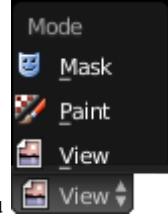
The rightmost group of four icons make a further breakdown of the image into RGB channels with/without alpha

transparency layer, alpha on its own, and Z (depth) buffer on its own. Alpha transparency only matters if you are rendering a transparent background in place of the sky, as discussed later.

3.1.2 Viewing/Editing Texture Images



Here is what the variable part of the window header looks like when you select some other image, or create a new



one. Now the editing-context menu becomes useful: selecting the "Paint" item lets you paint on the image.

Note the rightmost group of icons in the header has shrunk slightly: there is no more Z-buffer option, though the alpha-related options still exist.

3.1.3 UV Mapping



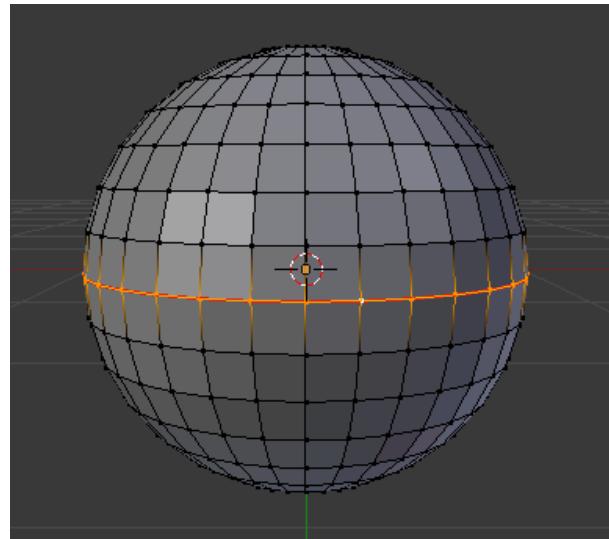
Here is what the variable part of the window header looks like when you select a texture image, and go into Edit mode on a mesh in the 3D View window.

3.2 UV Map Basics

In case you're wondering, UV mapping stands for the technique used to "wrap" a 2D image texture onto a 3D mesh. "U" and "V" are the names of the axes of a plane, since "X", "Y" and "Z" are used for the coordinates in the 3D space. For example: increasing your "V" on a sphere might move you along a longitude line (north or south),

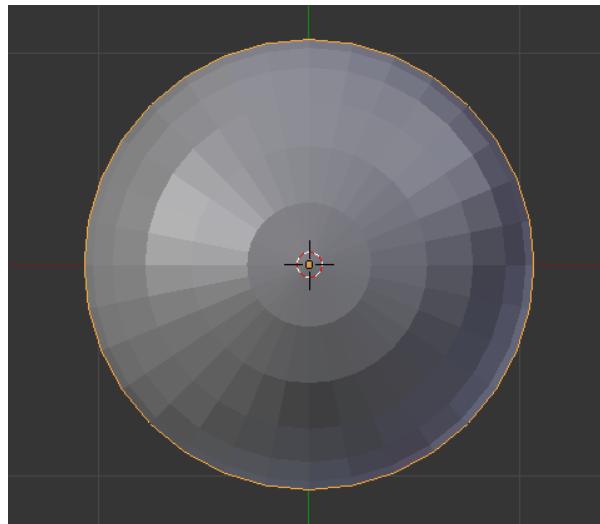
while increasing your “U” might move you along a line of latitude (east or west).

Another explanation can be gleaned from the Blender manual. Imagine a paper 3D model of an object, e.g. a sphere, that is to be laid flat on a table. Each of the 3D coordinates of the sphere can be mapped to the 2D coordinate on the flat piece of paper. Blender provides another view of the vertices (coordinates) in the UV/Image Editor. You can select and edit these 2D vertices just like in the 3D Editor window. The purpose of this unwrapping of the coordinates is just to map these coordinates to images/pictures so that the 3D image can have a realistic looking surface with textures derived from these images.

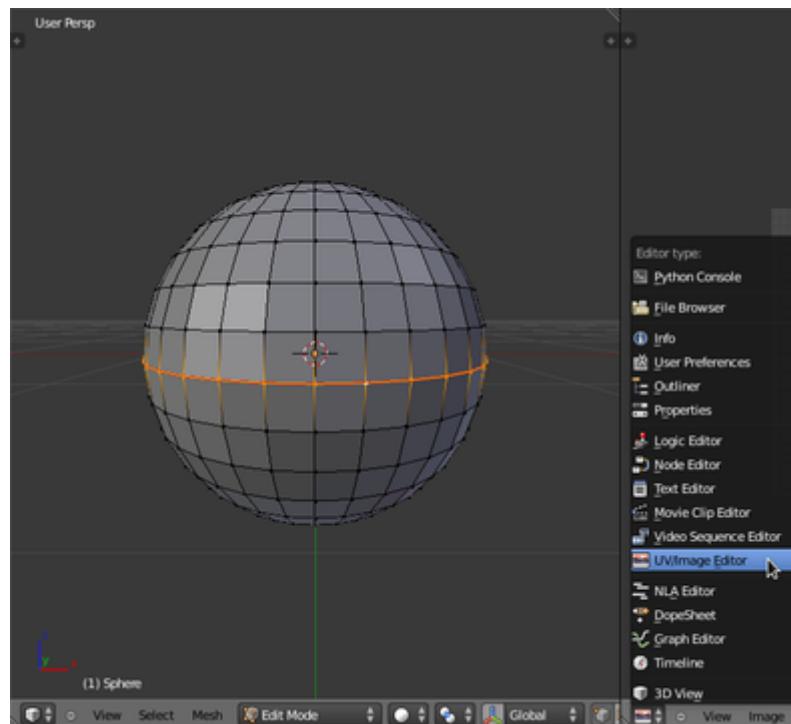


3.2.1 The Basics of UV Mapping

Add a sphere We'll use a sphere for this demonstration. Delete the initial cube (x). Then, create a new model, a sphere (**Shift+A → Add → Mesh → UV Sphere**). Leave the settings at default for now.



Unwrap the mesh Next, create a window for the UV mapping: click and drag left the small lined area in the top right corner of the 3D window, a new window will be created. Set its window type to “UV/Image Editor” with the drop down box at the bottom left corner of the new window or with Shift+F10 .

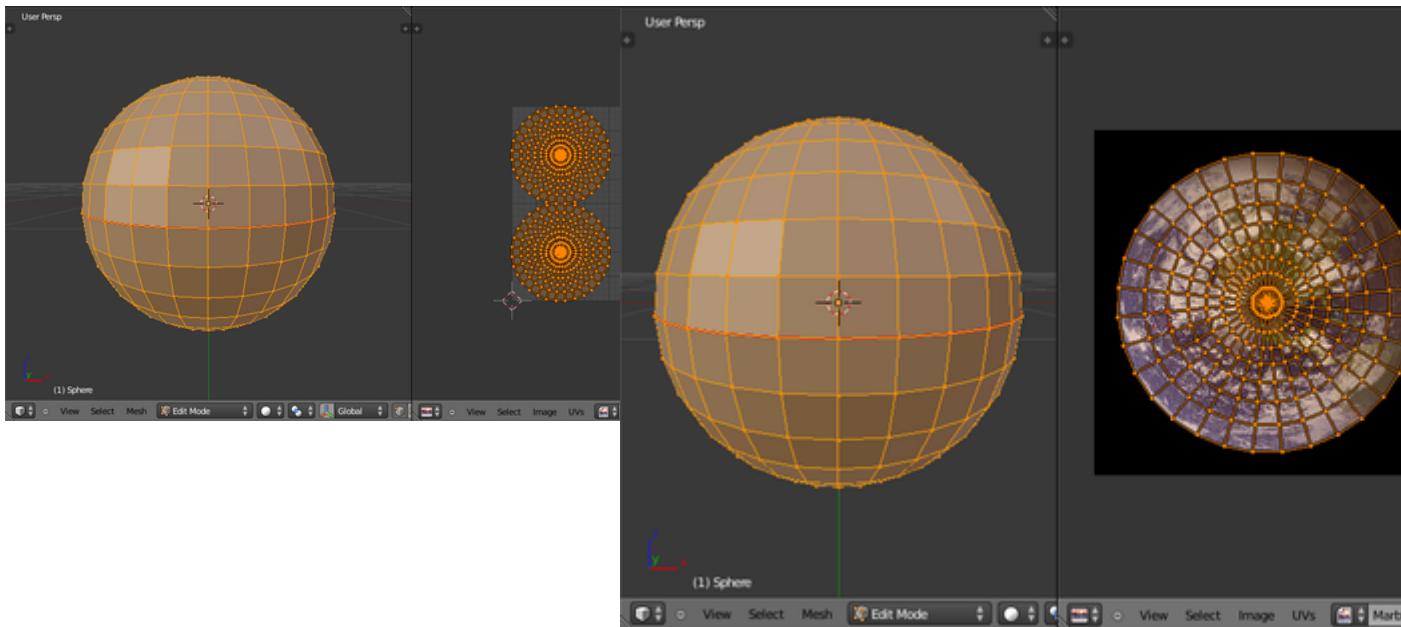


Mark a UV seam In edit mode, select a ring of vertices around the widest part of the sphere (the equator, if you will). This can be done easily by holding down Shift + Alt and right-clicking on the 'equator'.

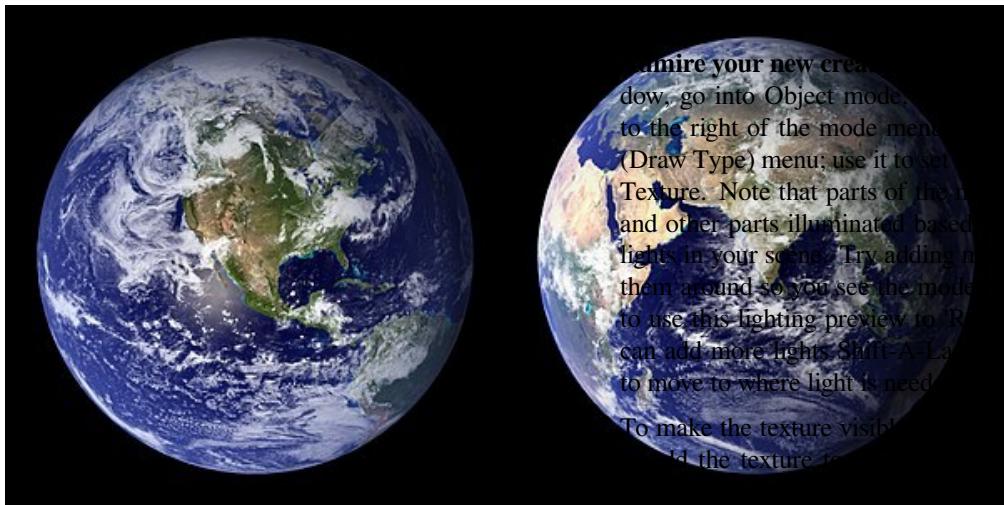
Press **Ctrl+E** and select *Mark Seam*, or select *Mesh → Edges → Mark Seam*. This tells the UV unwrapper to cut the mesh along these edges.

In the 3D View window, select all your vertices by pressing '**A**', and press **U → Unwrap**. In the UV/Image editor you should see all the vertices/faces/edges of the model represented, or 'mapped', in 2D space.

If you don't see anything, you need to select UV Editing from the Screen Layout dropdown in the UV/Image editor window.



Apply an image Now we're going to actually use this UV map. Save the following image (click to view in full high resolution ($4,096 \times 2,048$ pixels)):



Load it in the UV/Image Editor window by clicking *Image → Open Image* or by pressing Alt+O . Then with the grab, rotate and scale tools, adjust the UV islands (the UV groups that aren't connected to each other) so that it fits nicely on top of the image as shown. To select a UV island, hover the mouse over the island and press a vertex and L .

Note: Since the aspect ratio of the image will warp the UV's, it may be easier to simply re-unwrap the mesh exactly the same way you did before. You can then adjust the UV's as needed.

If you have issues fitting to the image use X and Y letters after pressing S. This will adjust the shape to fit the image.

[S]cale, [R]otate, [G]rab, [B]order selection, [A] Select/Deselect all

Admire your new creation! Open the 3D View window, go into Object mode. In the next drop-down menu to the right of the mode menu, open Viewport Shading (Draw Type) menu; use it to set the Viewport Shading to Texture. Note that parts of the model will be shadowed and other parts illuminated based on the location of the lights in your scene. Try adding more lights and moving them around so you see the model more clearly, and try to use this lighting preview to 'Reach' your lights. (you can add more lights Shift+A-Lamp and right click lamp to move to where light is needed)

To make the texture visible in renderings, you also need to add the texture to a material. In the Material context by

clicking the small shaded-sphere button. Create a new material by pressing the New button, leave the settings as they are for now; then switch to the Texture context. Create a new texture and select the type to be 'Image or Movie'. Select the globe texture from the dropdown menu.

Then go to the mapping menu and set the Coordinates to 'UV' and the 'Map' to 'UVMap', leave the projection at 'Flat'. This will make use of the the UV's we unwrapped earlier. To make the globe a bit smoother, switch to the Modifier context in the Properties window and add a subsurf modifier; set the number of subdivisions to 2. Then press T , if you had no Tool-shelf, in the 3D viewport and select 'Smooth' under the shading options. This will make the globe far smoother and more realistic.

Now, simply press F12 to render your scene!



Note that in the above render I've changed the lighting and the camera position to make the image more interesting. I also added a star background, you can find the settings for this in the World context (a pretty relevant coincidence) in the Properties window and the texture settings for the World.

Go to the **World** properties and check both **Blend Sky** and **Real Sky**, set all 3 colours (Horizon, Zenith, Ambient) to **black**. Go to **Texture** settings and if you weren't already there, click on the **World** texture settings (upper left globe icon) and set the type to **Voronoi**. Scroll down to **Colors** panel and check **Ramp** and give Color Stop 0 a white color and an alpha of 1 and Color Stop 1 a black color and alpha of 0. Now some will happen magic: set **Distance Metric** to **Minkowski 1/2**, in **Noise** section set size to **.002** and set 2 of Feature Weights at around **.5**. Finally, in **Influence** panel, check **Horizon**, **Zenith Up** and **Zenith Down**. That will do the trick. In some older versions there was a standard option for this but it has "disappeared".

3.2.2 Alternative method

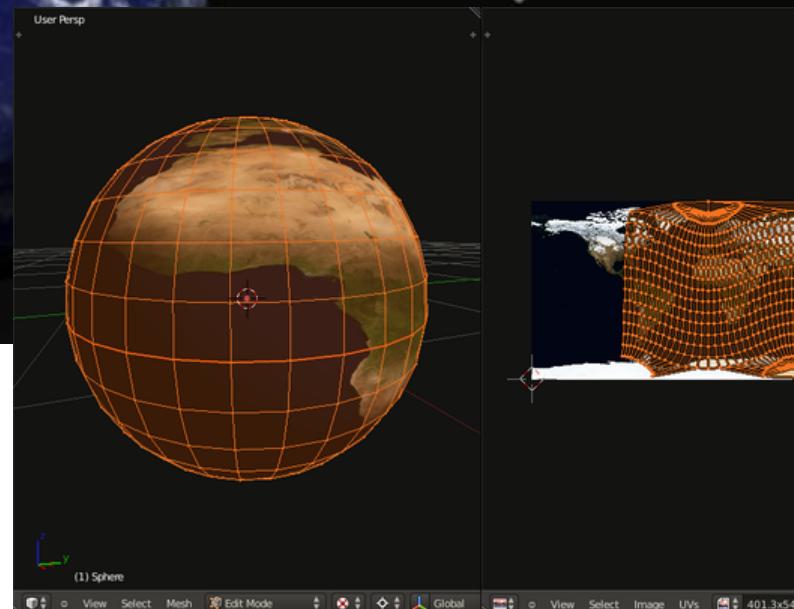
If you looked around the textured globe we just made, you would have noticed that around the 'equator' there were lines, or 'seams', where the two UV islands met. This is a common problem with UV mapping and there are a couple of ways to avoid it. In our case, since we're using a sphere, the best way to remove the seams is to use spherical mapping.

First download one of the textures from this link:

http://earthobservatory.nasa.gov/Features/BlueMarble/BlueMarble_2002.php. Then use this texture for your globe instead of the one we first downloaded.

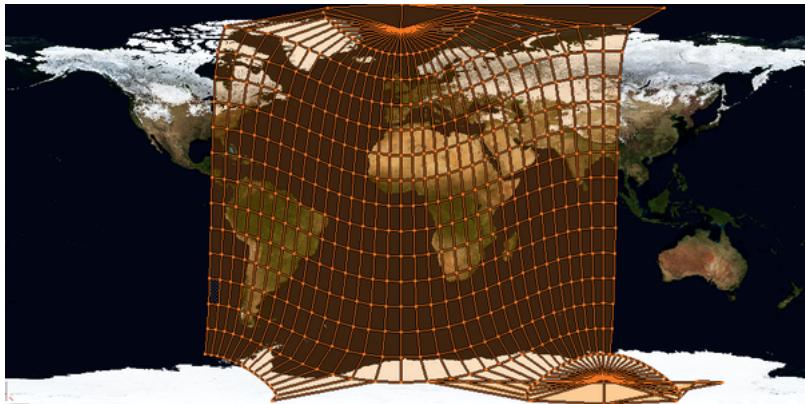
Go into edit mode, and select all the vertices of sphere. Press U → *Sphere Projection*. If you've got the UV/Image editor open you should see a very different UV map, as shown below. Note that only the 'Unwrap' option will use the seams we just made, all of the other options completely ignore it.

You can also try, especially handy when you use a more detailed map in the place of the seam: First unwrap with cube projection and then with sphere projection. It's now a square like UV.



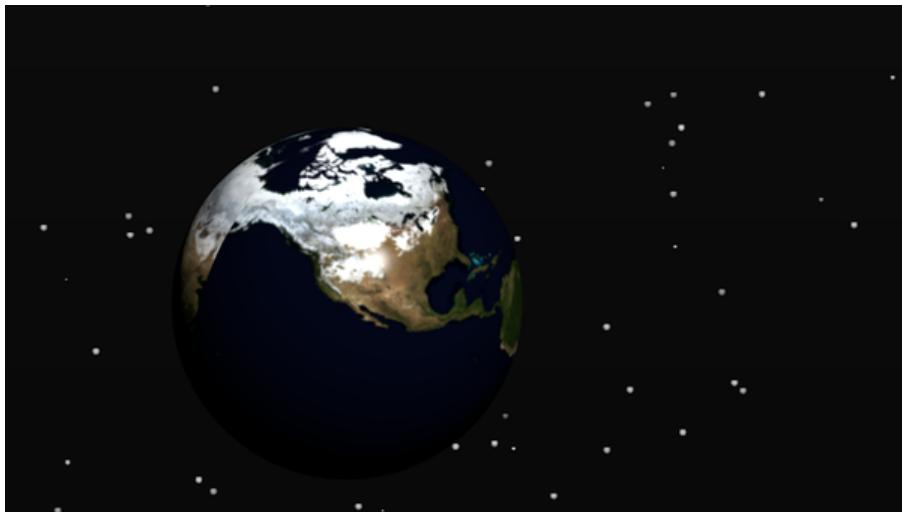
Now let's make the UVs fit more evenly over the texture. In the UV/Image editor, press **UVs** → *Constrain to Image Bounds*. This will make sure that during editing none of the UVs will go over the bounds of the texture; if they did, it would cause the texture to repeat itself in the area of the UVs outside the bounds.

Now select all the UVs with A and scale them up till it is stopped by the image bounds. In my case, there is one vertex in the top-right corner, and a few in the bottom-right corner that are quite distant from the rest of the UVs; to fix this, I simply selected those vertices and moved them closer to the rest of the UVs.



Now simply scale all the UVs by the X axis until the UVs stretch over all of the image. Doing this is exactly the same as scaling in the 3D viewport, press S and X , then drag the mouse.

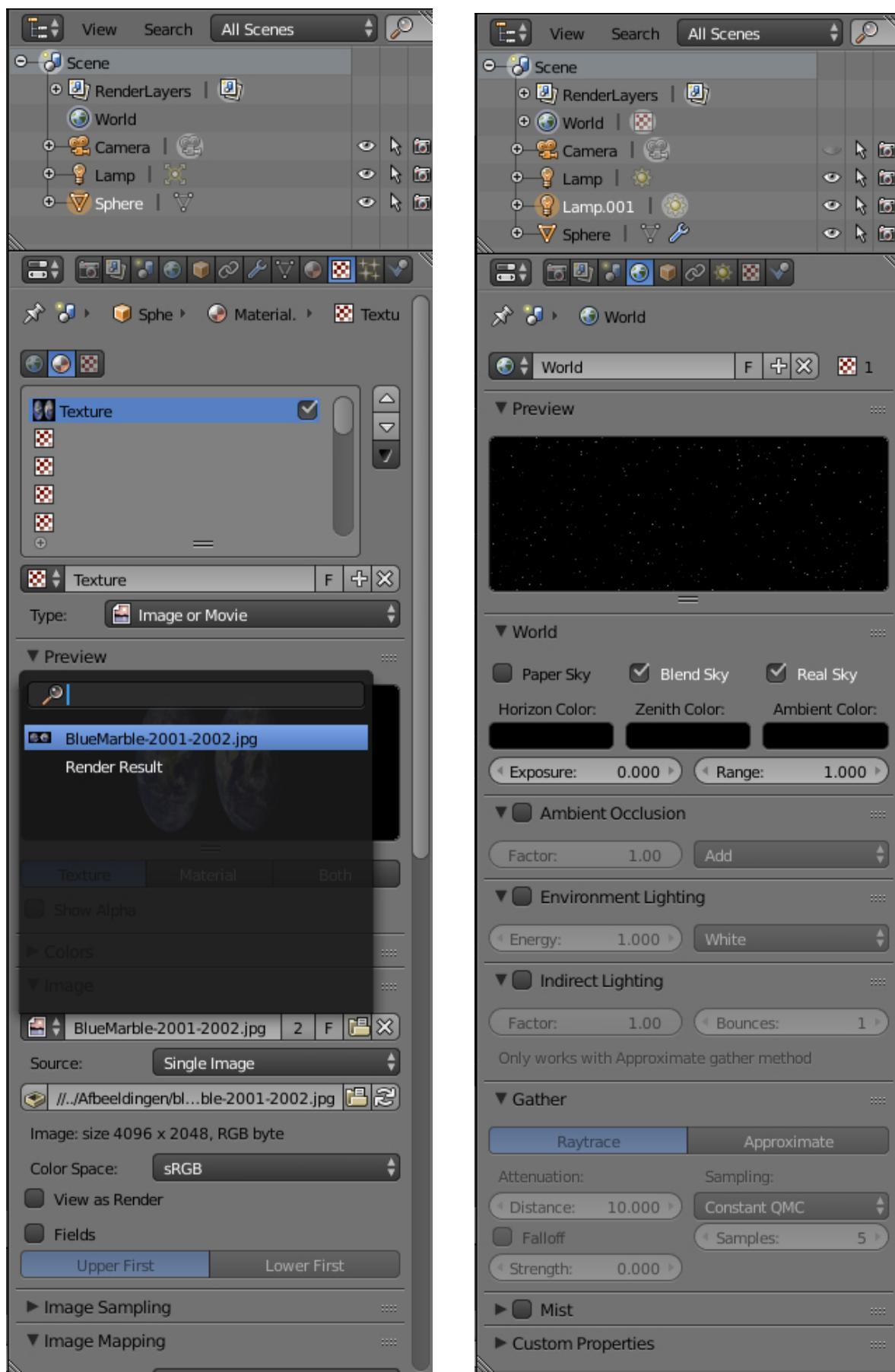
Now rerender your model, and you will see that there are no seams! You can also use an image with clouds. if you render this you will see seams. you can fake this away a bit by making the diffuse at material sea-blue by using the color picker after rendering the first render (If it looks good it is good).

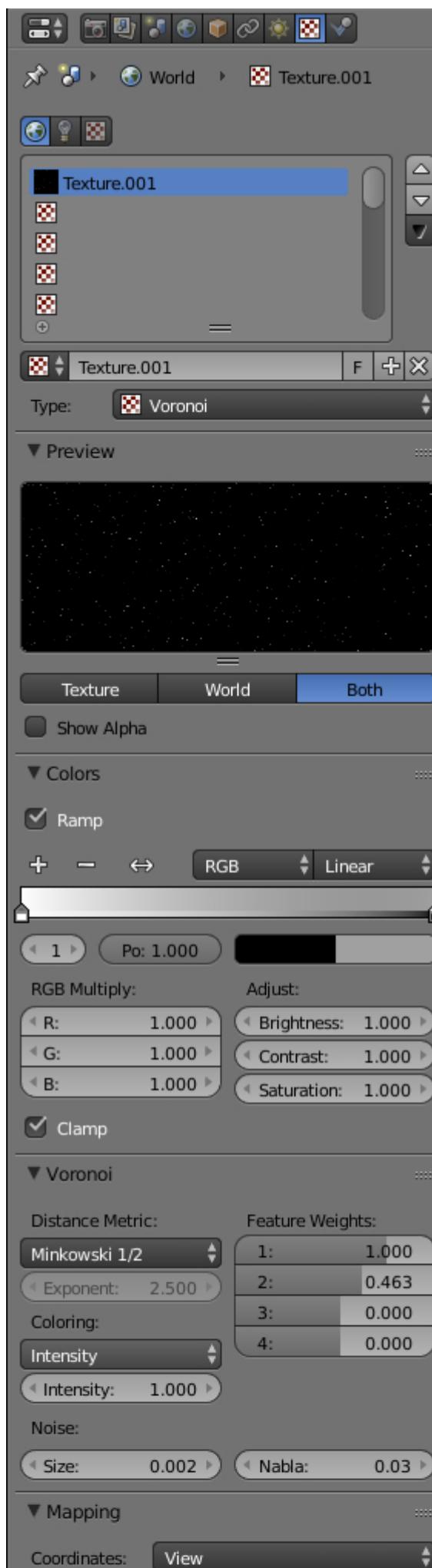


3.2.3 Learning More

In this tutorial we have barely scratched the surface of what is possible with UV mapping. Below are some more tutorials that will help you learn more about Blender's UV tools:

1. How to use multiple UV maps: <http://cgcookie.com/blender/2011/09/14/tip-using-multiple-uv-coordinates/>





3.3 Realistic Eyes In Blender

3.3.3 Building The Eye

Note: Some pictures are outdated.

3.3.1 Overview

NOTE: this tutorial is incomplete!

This tutorial will teach you how to effectively utilize textures and materials in Blender to create realistic eyes for characters. This tutorial will not teach you how to finish the eye. Because you did already learn that in previous tutorials. This tutorial was inspired by a great Maya tutorial in the Gnomon Workshop series by Alex Alvarez. My goal with this tutorial is to teach you how to get those same stunning results using Blender. Please check out Alex' tutorial as well, as it is very informative and covers more detail than I will be covering here. This tutorial assumes you know your way around the Blender interface and so I will not explain each key command as I go, but will instruct you in what steps to take and point out details such as key commands only when it seems relevant to do so. In addition, you will need some sort of image manipulation software, as this tutorial relies heavily on image-based textures, which we will be preparing outside of Blender. I will be using GIMP, but feel free to use Photoshop or whatever you're comfortable with.

1. The 4 original Maya tutorials in video format []

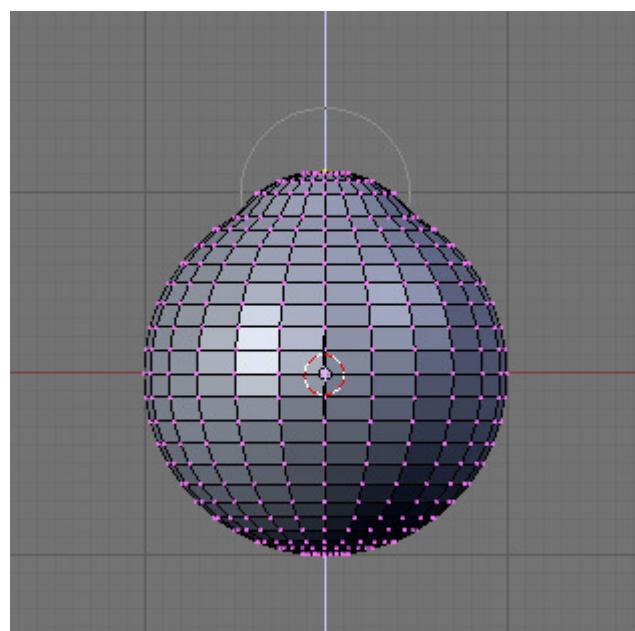
OK, let's dig in...

3.3.2 Reference Material

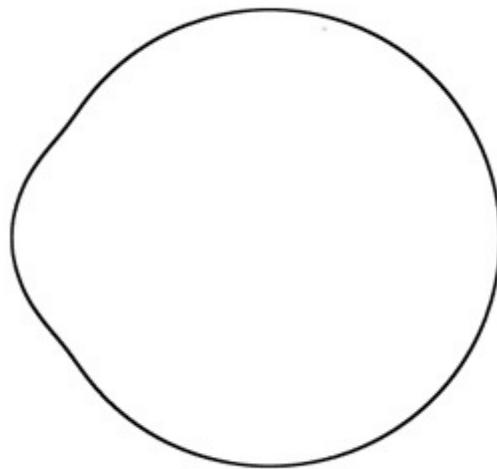
Let's Take a moment to familiarize ourselves with the anatomy of the human eye. Do a Google image search and examine whichever diagram is to your liking. Now, you don't have to pay attention to every single feature, but do take note of the **cornea**, the **iris**, the **pupil**, and the **sclera**. These are the only parts that are visible to us when we look at a person's eyes, and therefore the only parts we will actually need to create. Many of the diagrams you find out there will be disproportionate, and so in order to know how large to make our features in relation to one another, it is best to view a large number of diagrams, a large number of eye photos, and also photos of plastic models of the eye, as plastic models tend to be truer to scale than 2D drawings. I also found a photo of a guy popping his eyes half-way out of his head. All of these together gave me a good feel for what the appropriate proportions are for a human eyeball. To some extent though, cornea size and sclera shape will vary on a person-to-person basis, so there is no precisely correct size, per se.

Creating a UV sphere. 24 segments, 24 rings, and a radius of 1 is recommended. The top of the UV sphere will become the cornea of the eye. So now let's go into **side view**, in the **orthographic** view, and with the top-most vertex selected and **proportional falloff** enabled, **grab** and raise this vertex along the **Z axis**. Don't click to finalize the edit just yet - first we will use the **mouse wheel** to control how much falloff is going to be applied. This process can require some finesse and multiple adjustments until the bump of the cornea is anatomically correct. You will want to set the falloff shape to "**sphere**". You can also make 2 balls, like in the Procedural eyeball tutorial, and let the iris bend inwards of the inner ball.

If you like you can use a Torus for the iris, this is difficult but fun to play with.



Below is a render of the curve I ended up going with. You can use it as a guide when modeling if it helps you. *This is with a SubSurf of 3 applied*, so keep that in mind.

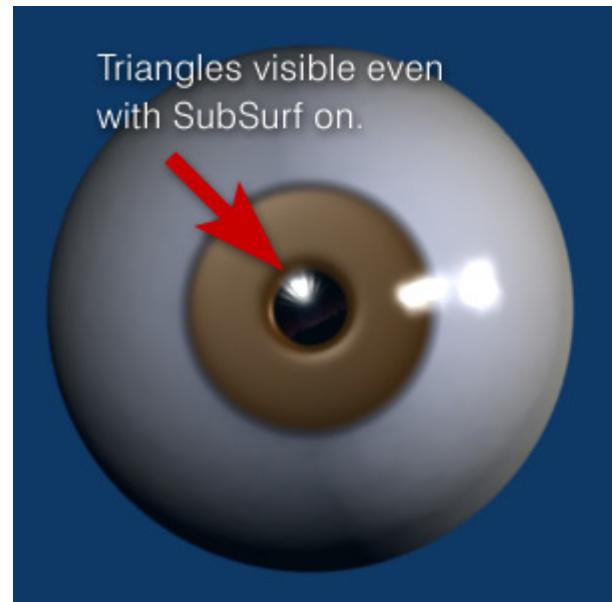


Now, some people may prefer to use a NURBS sphere for the geometry of their eyeball. If you are comfortable with NURBS modeling, and texturing NURBS models, go right ahead. It will give you a smoother and more rounded eyeball. I prefer not to complicate things with NURBS when polys will do just fine, and am guessing most of you feel the same, so I'm sticking with polys. This does however bring us to the topic of the SubSurface modifier and how it will affect your eyeball. When SubSurf is applied to a UV sphere such as our own, it causes the previously spherical shape to become slightly oblong. In other words, the eye will become nearsighted. This is barely noticeable, but it does happen, as a result of the way SubSurfing affects a UV sphere. You could create a UV sphere with more rings to ease this if it bothers you. It looks fine for our purposes though, to be honest. See the comparison in the animated GIF below.

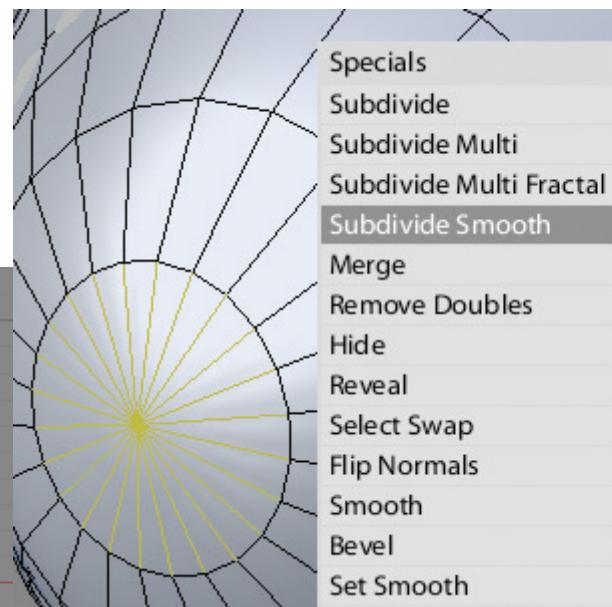


There is one last issue to address with regard to using a

polygon-based eyeball. That is the small triangles that persist on the poles of the eyeball. These stubborn tris will remain and you will find that they result in an ugly and unnatural pinched effect when a specular highlight appears over the center of the cornea. Adding a SubSurface modifier will not make them go away. This didn't become apparent until after I did some texturing work and test renders, but I want to point it out to you now, before we get into all that. (See examples below)



This is a screenshot from the 3D Viewport showing the "Shade Smooth" option placed near the cornea.

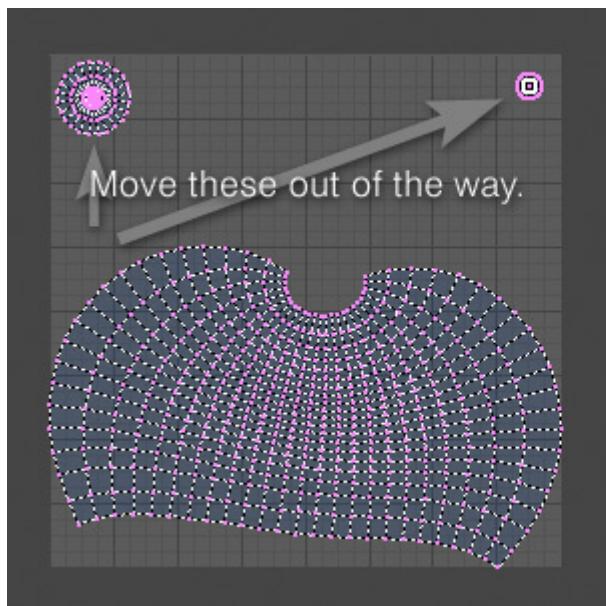
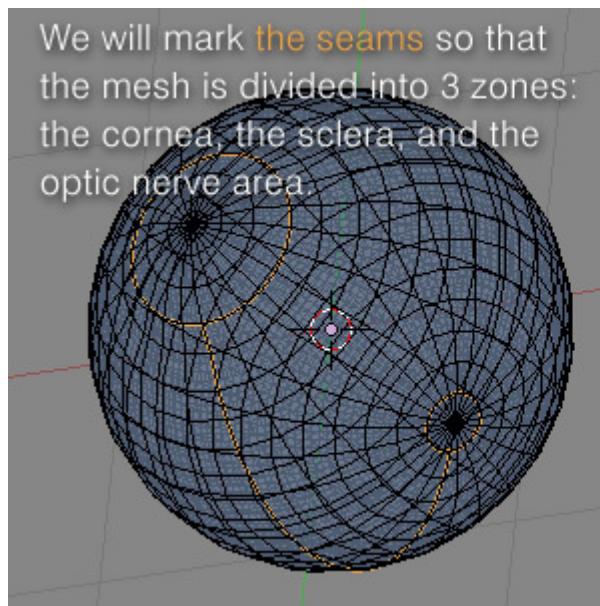


Now repeat this process at the opposite pole of the eye (the optic nerve area).

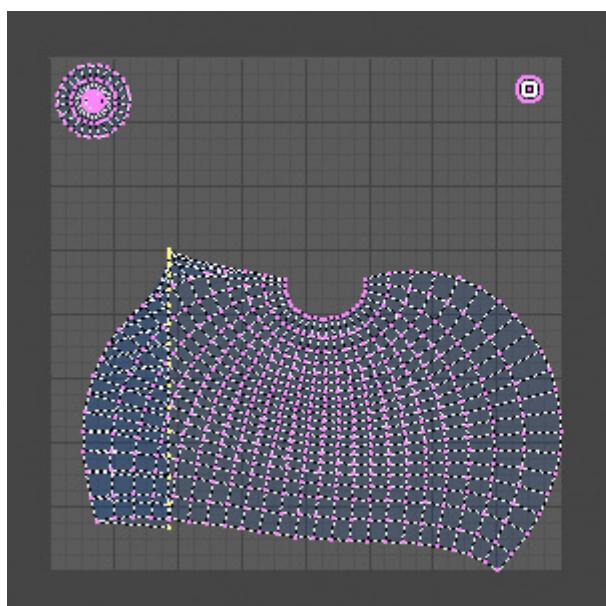
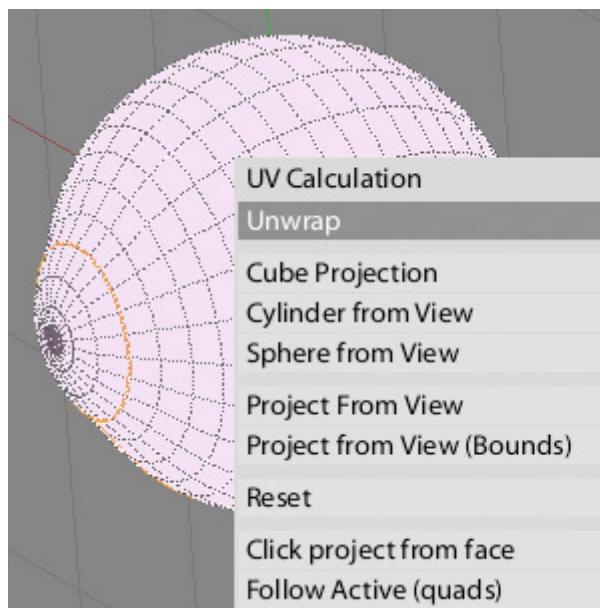
We are now ready to begin texturing the eyeball.

3.3.4 UV Mapping

In **Edit Mode**, we will define the seams for our eye. Select the edges you want to mark as seams and type **Ctrl E**, then choose **Mark Seam** from the menu.



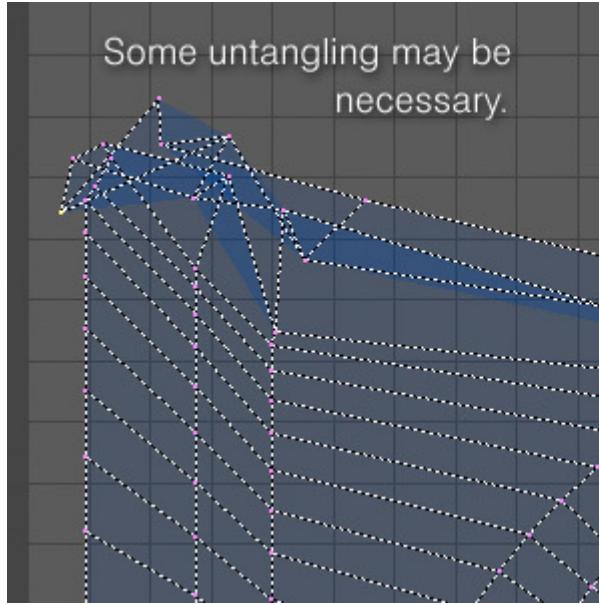
With all of your mesh selected, type **U** to unwrap. We can now see our unwrapped UV layout in the **UV/Image Editor** window. (For older versions of Blender, you might need to select **UV Face Select** mode to unwrap, but it's better if you download the latest.)



Constraining to the **X** axis, by typing the "**X**" key after you grab, is a good habit to get into while straightening these out.

We will make the iris on another mesh so don't spent time on the cornea UV island.

Those two islands we see are the cornea and optic nerve area. We can move them out of our way by selecting any vertex on them and typing **L** to select all linked vertices. Then **G** to grab as usual.

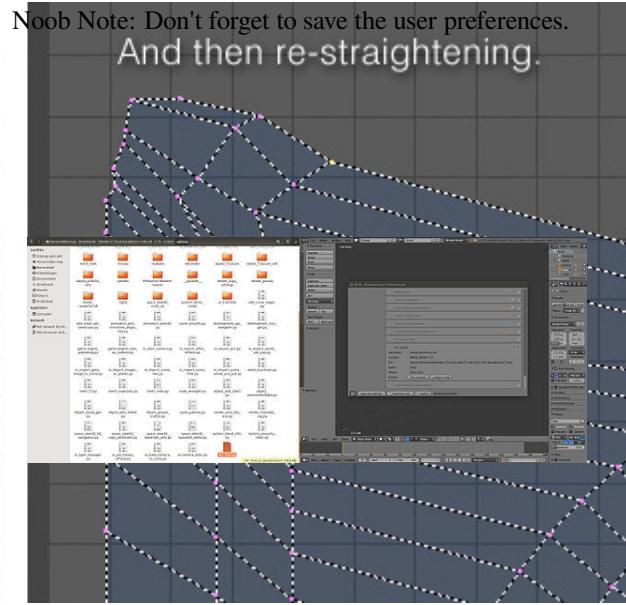


Straightening out all of these vertices into a usable grid shape will take you some time so we won't do it that way. Having a rectangular grid will prove very helpful when we are texturing our eyeball. so we will use an Add-on for blender. You'll want to use Blender's grid as a reference and just align the rows and columns to it exactly. Yes, you can get it *exactly* aligned, since there is a snap-to-pixels feature. Click on snap-to-pixels in the drop-up menu in the uv/image editor header. (Don't move the 2 circles when snap to pixels is on cause that will ruin them.) You can scale the whole thing up later, once each row and column is evenly spaced. If you can, leave some space between your UV layout and the border of the layout area, if you "bleed" it to the edge, you may have texturing problems later. But you can scale it down later, so no problem. we will download the UV Tool from Daniel Banasik. A wonderfull free tool which can smooth an align the vertex loops to a beautiful grid with a little effort. The old method was aligning every single vertex 1 by 1 :(.

[Page of the tool \[\]](#)

Installing an Addon is very simple. If you've like me not installed blender but use a standalone version it's done in no time. All you have to do is put a folder or file in a folder blender is reading for add-ons.

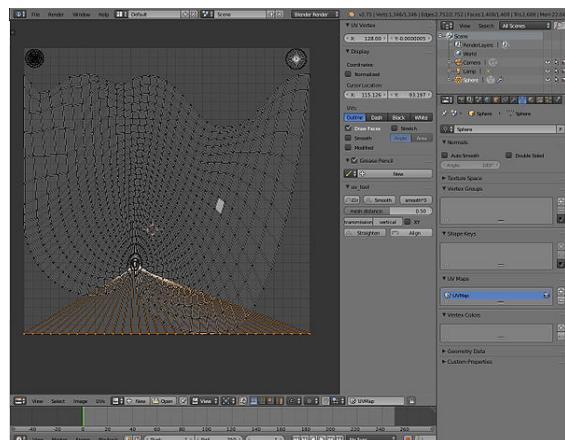
Just download it and Unpack it. What you've unpacked is, most of the time, precisely what you have to put in the add-on folder. In the blender standalone version folder (if you don't use a standalone version of blender, it's Operating system specific how to place an Add-on.): go to the folder with the name of the Version Number (at my version it is 2.75)-> scripts -> add-ons. then put it here. Save your work and quit blender. Then restart blender and open your work. Go to User Preferences under File in the blender program header (the uppermost) and go to addons. Go completely down and open the details of: UV: uv_tool. Then click the checkbox: most right and you're able to use it.

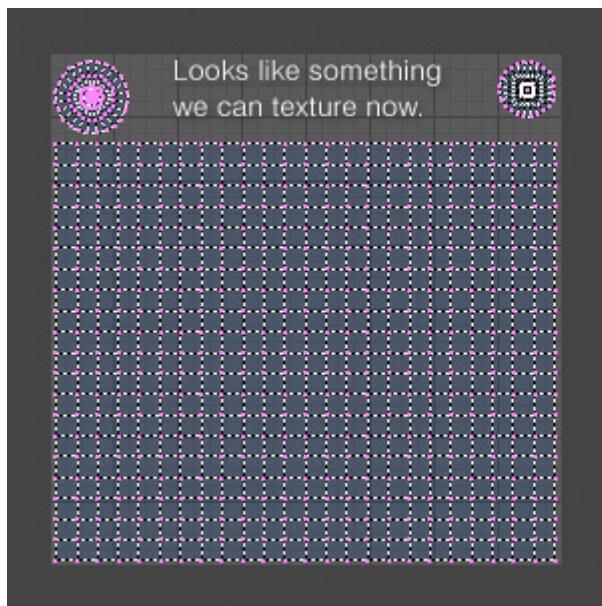


If you, like me, love Disney-Pixar Movies you can recently download an Uncommercial (you're not allowed to earn money with it) free version of Renderman, the program Disney Pixar uses to render their movies. You can install it and use a third-party add-on (PR-man) to implement it in blender. But that will come in later tutorials.

How to use:

select the loop on the right side and click on align, do the same for the left. You can't do more than 1 loop at a time otherwise you'll go out of Edit Mode in the 3D View. Then go to 3D view put back in Edit Mode and go back to UV/Image Editor. You'll Also notice nothing has aligned. so do them loop for loop. Just select the loop and click on Align. It will be automatically placed correctly. Just move on till they're all done.





It's important to pay attention to which way the UV layout is facing. Chances are the top of the sclera region is actually what you will want to be the bottom. It doesn't technically matter which way it's facing, but you will probably want it to look intuitive when you are painting the textures.

You can check this by going to the 3D view (or having split windows so you can see the 3D and UV views at the same time) and making sure everything is selected, then deselect 1 vertex close to the optic nerve or the cornea. Back in the UV/image editor the vertex will disappear all connected edges and maybe more from your UV map, so you can work out how the map corresponds to the actual mesh surface.

Once you're done, **save your .blend file** and then **save your UV Face Layout**. This is done in the UV's: **Export UV Layout**.

3.3.5 Preparing A Template For Texturing

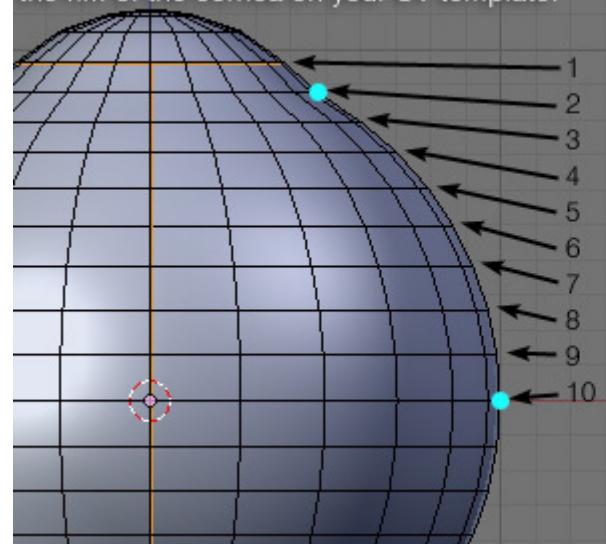
Now launch your image editor and open the UV face layout you just saved. (You probably saved it as a .png file, since that is Blender's default format for UV layouts) So we are now looking at a UV grid with the square and the 2 islands.

Open gimp and rescale the image to 4092 by 4092.

One other element that will be useful to have in our template is a layer of markers that denote where the equator of the eye is, because what looks like the middle on the UV layout is not actually the middle. To find the middle, go back into Blender and count the rings of your eye mesh, starting at the seam of the cornea. Once you get to the equator, take note of how many rings it took to get there. Go back to your image editor and mark that area. You can mark it with a guide, or mark it with a coloring of some sort - as long as it's clear to you that it indicates the

equator of the eye mesh. It may also be useful to mark the point where the transparency of the cornea should begin.

But it's upside-down!
Count the rings, then mark the equator and the rim of the cornea on your UV template.



Now we will make the texture. To begin make the inside of UV square and islands white, a neutral colour.

Do this by selecting (round or square) the UV island or square and then clicking 2 times on the bucket and set the Affected area to Fill whole selection then switch the black and the white colour by clicking on the arrows in the toolbox (If you've no toolbox click on tools in the program header and click: new toolbox) and click with the bucket on the selected area and do this for every part of the eye.

an eye is reddish at the back so make the back point directly reddish, I use FB2F05.

Select the back point so you can fill it with this beautiful colour.

Select the square so you cannot "Spill" over the corners. We will paint the back with the same colour of the back point 'till over the middle of the eye. click the airbrush tool and increase the value of the size and also the flow if its a very transparent at the end it has to be very transparent.

Noob Note: make the selection a little bit bigger then the square or the UV island to prevent seams.

Search for a nice iris texture:

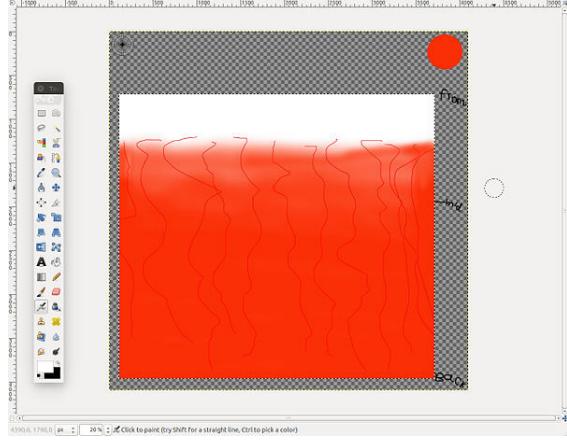
1. Google: iris texture []

Make sure you have around 1024 by 1024 pixels.

Open with gimp and crop it out by pressing image -> Crop to selection save it as .png and use it later. If it's a little bit to small, don't worry you won't render it that big.

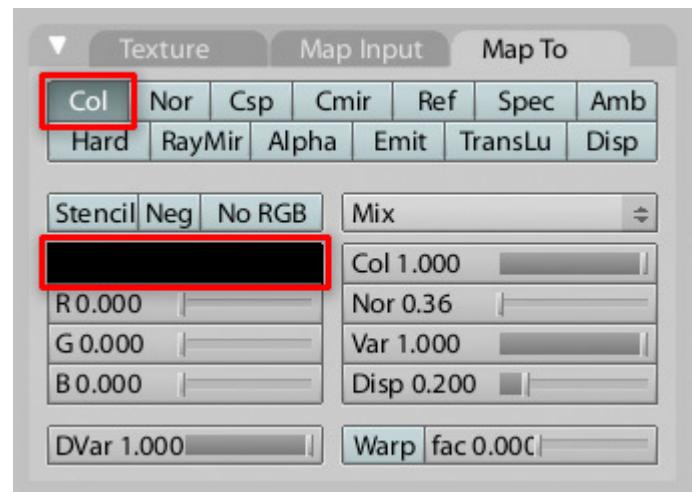
If you like you can make blood vessels in the cornea with the pencil tool and the colour FF0000. Make sure you shrink the to about 10. Only don't try to copy my vessels

cause they are to bad.



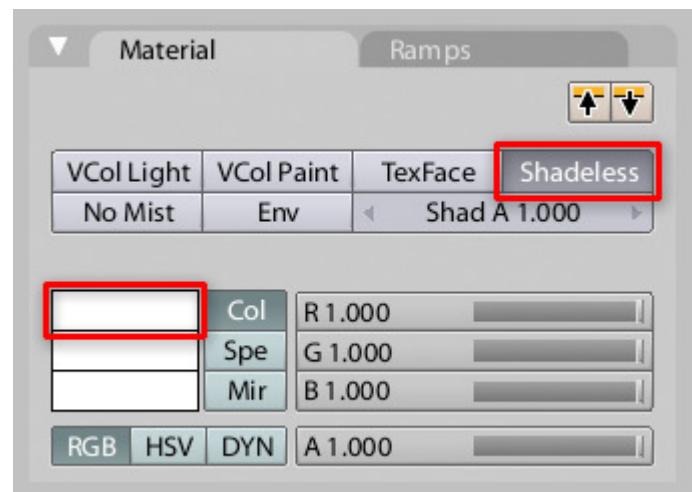
You can design your own texture to apply.

Save your texture as .png.

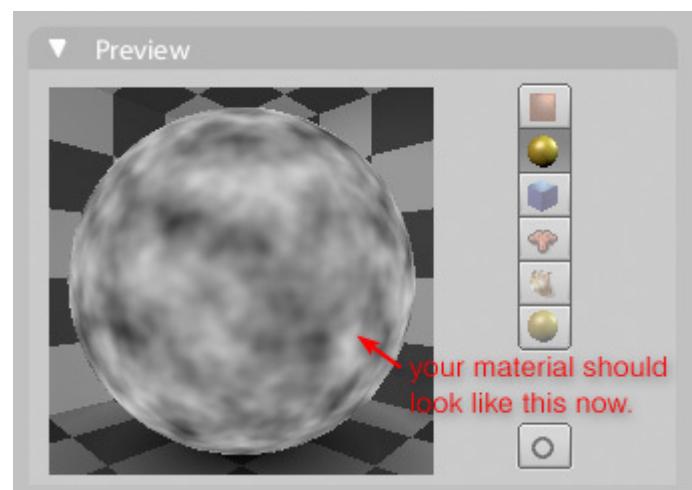


Now go to the **Material** properties. Change the diffuse color to **white** and click the **Shadeless** checkbox.

Note: In this particular case it doesn't actually matter which of these colors is set to white and which is set to black. The end result is basically the same, and you can always invert the image later.



Check the Material Preview window to make sure it looks right.



Time to bake your eye. Make sure your eyeball object is

3.3.6 Bump Map - The Sclera

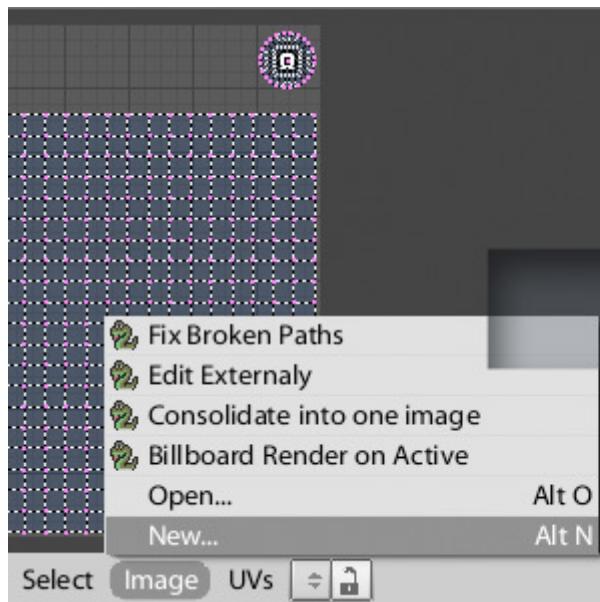
If you want to use a bump map in your eye.

Create a material for your eyeball. Then within that material **create a texture**. Name this texture something like "*BumpMap(Procedural)*". Set the texture type to **clouds**. A noise size of **0.25** with a noise depth of **2 or 3** is fine, and you can experiment with other values if you like. The other texture settings can be left at their defaults.

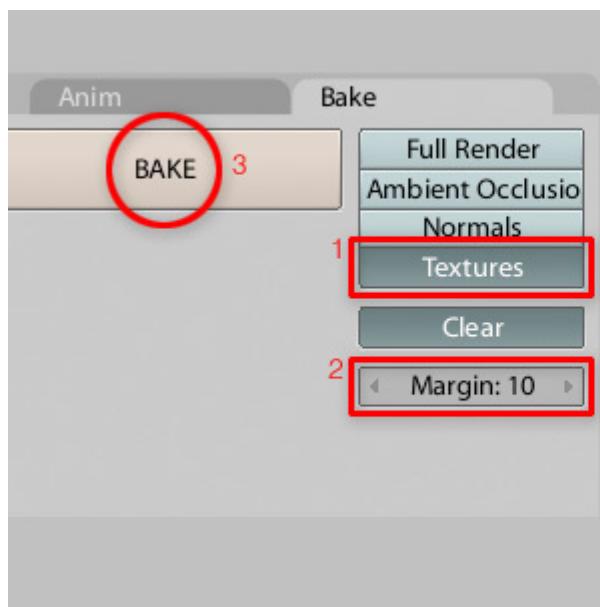
Now we are going to bake this procedural texture to get an image-based UV texture that we can edit in our paint program. Why? Because there is not an even level of bumpiness around the eyeball. The sclera has bumps, while the cornea doesn't. If your cornea was bumpy, your vision would be distorted. We want our 3D eyeball to accurately convey this subtle difference when light is reflected from the surface.

Before baking the texture, there are some parameters we must specify. In the materials panel click the **Map To** tab. The Color button is depressed by default, so you can leave it as is. Make sure this is the only depressed button. Then **change the color to black**.

selected and go into **UV Face Select Mode**. Then **split your screen** and set one viewport to the **UV/Image Editor**. From the menu at the bottom of the UV/Image Editor, choose **Image > New**. Enter a name like "*Sclera-BumpBake*" and set both the Width and Height to **2048** (to match your UV guide). Click "OK" and a new all-black image will appear behind the UV grid.



Go to **3D view** then in the render properties to **Bake** and click **Textures** at the **Bake-mode** drop down menu. Then set the margin to 10 and click the **BAKE** button. You will see the black image in the **UV/Image Editor** fill in with your bump texture. From the menu choose **Image > Save As Image**.



Now save your .blend file. You are able to make the rest of the eye yourself and if you forgot, read to previous tutorials.

3.4 Beginning Lighting

Lighting, you say? Pssh. Just throw up one light source and let her run, right?

Lighting is probably the most underestimated part of a scene by new 3D artists. Unlike real life, it's actually quite easy to produce a uniformly well-lit, shadowless scene in Blender. But such scenes are usually pretty dull and boring.

Far more interesting are ones with highlights drawing attention to some areas, and shadows in other areas, leading the viewer to wonder what they might conceal. If a picture tells a story, and a picture is worth a thousand words, then lighting is like the punctuation for those words, the pauses between sentences and the sudden dramatic changes of tone.

Extremely uniform lighting can give a scene a "photoshopped" look, as though the artist took part of one picture and stuck it on top of another. Whereas even a little bit of shadowing of one object by another can reinforce the impression that they are very much three-dimensional objects with a definite spatial relationship to each other, located in the same scene.

The following tutorials will help you gain knowledge of the technical use of lights in your scenes.

3.5 Understanding Real Lights

In order to learn how to light a scene properly it is helpful to first learn a few things about real lights. When you add a light to your scene you need to understand how to get the best results by simulating the properties of the real light source that you want your Blender light to replicate.

Realtime Texture Mapping

3.5.1 The Properties of Real Lights

Pack Image as PNG
Pack Image

Every light source always has a distinctive color of its own. A photograph taken at sunrise or sunset is very easy to distinguish from one that is taken at noon just by looking at the color of the light. Incandescent light bulbs emit a light of a color that is different from that of fluorescent light bulb. So a scene lit with one of the two will have a different color cast when compared to the other.

Angle

Light always streams in from a particular direction. The angle or direction of light has a particular influence on which planes of an object receive light and the shape and direction of the shadows that are cast by objects receiving light from that particular source.

Brightness and Decay

As light travels through space it decays or grows dimmer with distance. Think of a firework bursting in the night sky. When the firework first bursts the sparks are bunched close to each other. Over a short period of time they quickly spread out from each other. If you replace sparks with photons of light and the firework with your light source than you have a very rough analogy to how and why light decays. All real lights decay with distance according to the inverse-square law. At times when lighting a Blender scene it becomes necessary to cheat this law.

Shadows

Real lights will always cast a shadow of some sort. Real scenes also typically include multiple sources of light (even if these are only *indirect lighting* as a result of light bouncing off other objects). Thus, even in an area shadowed by one light source, there will typically be light coming from other directions, to soften the shadow so it is not completely inky black.

Size

Real light sources rarely take the form of tiny points; the light usually comes from an area of discernible size. This manifests itself in the form of a softness to the edge of the shadows, or *penumbra*: areas behind an object where the light is partially obscured are less dark than the inner parts of the shadow where the light is completely blocked.

3.5.2

These are not all the properties of real lights, but as far as CGI lighting is concerned they are the most important. As we will see later on, Blender offers enough controls for you to tweak and adjust each of these key properties of light.

3.6 Understanding Blender Lights

Blender provides several different kinds of lights:

- **Point**: Single point light source. Useful to provide very localized light. The shadows can be sharp, or you can set its size to something nonzero to make the shadows fuzzy. Can represent light sources within the scene; e.g. if there is a lightbulb or candle or something in the scene, position one of these within it to give the impression of light coming from that object.

- **Sun**: A light with parallel rays that will illuminate the scene with an even light. Because the sun is (effectively) infinitely far away, the position of this lamp does not matter, only its direction. Good to use in brightly-lit outdoor scenes (i.e. a sunny day).

- **Spot**: Spot lights produce light constrained to a cone-shaped beam, and have some special features. They are the only light source that can be made visible with the 'halo' option, to simulate light in a fog. They are also the only light source that can cast buffer shadows (see below).

- **Hemi**: 180°-wide uniform, shadowless light source. Great for use as a fill light, or as a back light, or to represent light from the sky. Similar to Sun, its position does not matter, only its direction.

- **Area**: These are similar to point lamps, except that they are rectangular. As a result they can cast accurate raytraced soft shadows, at the expense of additional render time.

There are two different kinds of shadows that lights may cast: *buffered* and *ray-traced*. The main difference is that buffered shadows are much quicker to calculate, but take more memory, and can be of lower quality without some fiddling. Also strand-rendered materials (as can be used for hair or fur) cannot cast ray shadows with the Blender Internal renderer, so you have to use buffer shadows for them so their shadows look realistic.

Only spot lamps can cast buffered shadows. Hemi lamps cannot cast shadows at all.

3.6.1 Light Settings

Blender's lights offer many settings you can mess with to adjust their effect, such as:

- **Energy** — the strength of the light.
- **Colour** — real lights often have a colour, rather than being pure white. It is common in photography to talk about lights adding "warm" (reddish/yellowish) or "cool" (bluish) tints to a scene.
- **Falloff** — real lights get weaker with distance according to the well-known *inverse-square law*. But Blender doesn't force you to conform to reality: point and spot lamps also allow for other options, like *inverse-linear* or even a custom curve.
- **Distance** — even with inverse-square falloff, the intensity of real lights never quite goes to zero at any distance, though it may become too low to measure. Blender's distance specification allows you to limit the effect of the light to a finite maximum distance. This can be useful for having multiple lights illuminating different parts of a scene, while minimizing

unwanted interactions between them—a headache that real photographers and lighting technicians cannot avoid!

- **Negative** — instead of *adding* light to a scene, this light source can actually *darken* the scene. Again, not possible with real lights, but useful for certain kinds of effects. In live action black cards are sometimes used to reduce reflection on the subject by absorbing reflected light. This light could serve to provide a similar function.
- **This Layer Only** — this light only illuminates objects on the same layer(s). Another useful way to minimize unwanted interactions between different lights.
- **Specular** — uncheck this box to prevent this light illuminating specular parts of materials.
- **Diffuse** — uncheck this box to prevent this light illuminating diffuse parts of materials.

Lighting Without Lamps

It is possible to light a scene without lamps, or with fewer

lamps. In the World Context  of the Properties Window, there are three options, for “Environment Lighting”, “Ambient Occlusion” and “Indirect Lighting”.

Environment Lighting adds a shadowless light that seems to come from all directions and fill all parts of the scene.

Ambient Occlusion (“AO” for short) is supposed to mimic the effect of shadows darkening corners and crevices of real-world objects (in theory this should naturally fall out of accurate lighting calculations, but it is easier to compute it separately); Blender also allows you to use AO to *brighten* parts of the scene outside those corners and crevices.

Indirect Lighting tries to mimic light bouncing off diffuse surfaces and illuminating other diffuse surfaces. It only works when the “Gather” option (next panel down) is set to “Approximate”.

- Ambient Occlusion Video Tutorial: <http://www.youtube.com/watch?v=rKUAemD7oo4>

3.7 Basic Lighting Rigs

Open a new Blender document. Delete the default cube, and insert the Monkey (Suzanne) instead. Hit F12 to render. You should get something like this.

See how the character’s face—arguably the most important part you usually want to look at—is in shadow? That is absolutely terrible lighting.



Blech

In this section, we will look at how to improve the lighting of this scene. The object of the game is not to get rid of *all* shadows, because flat-lit scenes tend to look pretty boring, too. Instead, we will look at how to make imaginative use of the placement and strength of highlights and shadows, to add interest and realism to the scene.

In photography, there are basically two kinds of lighting setups: outdoor and indoor. In the real world, outdoor lighting (at least in daylight) is dominated by the Sun. This is a single, extremely strong light source. But there is also indirect sunlight reflected off other objects, including the sky, and these tend to soften the shadows and even add some colour to them. Outdoor close-up model photo shoots also frequently make use of metal sheets, held up by support crew, to deliberately add more of this indirect reflection and make the lighting of a model more even.

Indoor (studio) lighting setups are commonly described in terms of the number of lights employed, commonly “one-point”, “two-point” or “three-point” for 1, 2 or 3 lights.

In 3D graphics, you can cheat over this outdoor/indoor distinction. After all, the Sun is just another light source you can choose to place in a scene. Instead of metal reflectors, it is usually simpler to just add more lights, even if it is meant to be an outdoor scene. In Blender, the lights themselves need not show up in the render, only their illumination of the scene.

3.7.1 One-Point Lighting

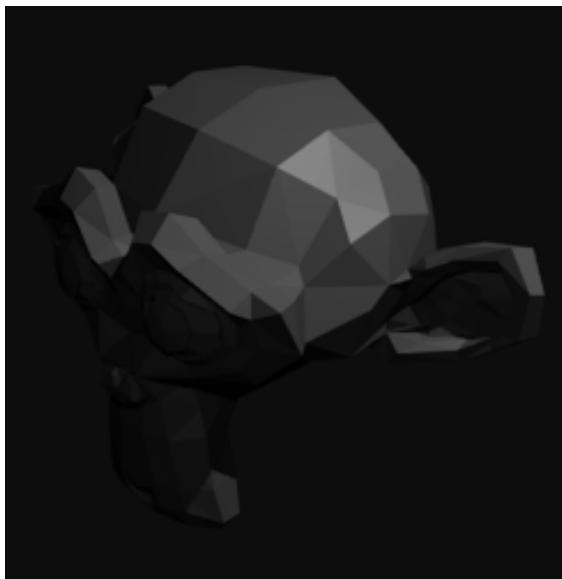
Continuing on from the above example, move the default lamp so it is roughly in the same location as the camera. Now your render should look something like this:

This is similar to the lighting you get when you take a picture on a point-and-shoot or cameraphone with flash enabled: because the light source is close to the lens, you don’t see many shadows (think about it: the parts the light

*Ho-hum*

doesn't reach are close to the parts your vision doesn't reach), leading to a very flat image. This is why experienced photographers often try to avoid using the flash.

3.7.2 Two-Point Lighting

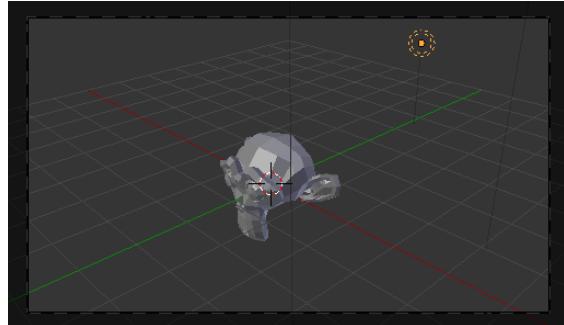
*At least some variation...*

This time I have gone back to the default light position, and added a second light (“fill light”) close to the camera, reducing its strength to 0.5, while the original “key light” stays at 1.0. That way the second light fills in the shadows just enough to make things legible, without flattening out the lighting completely.

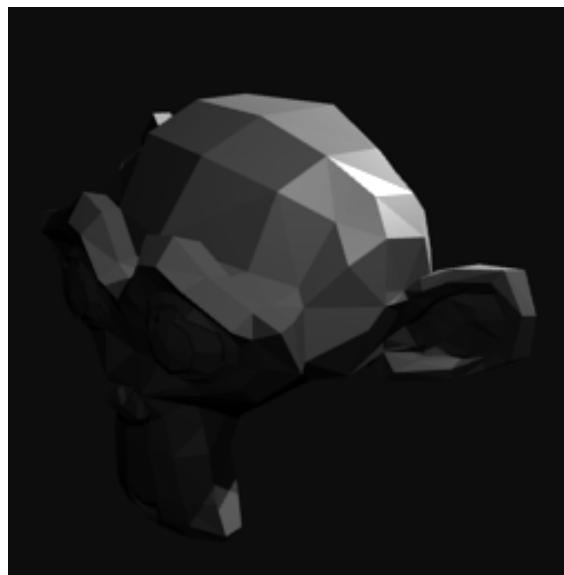
Set the direction of the light using object rotation. The light must be selected, press the **r** and move the mouse to change the angle.

The important point is that **the key (brightest) light is not at the camera position**. That way, the image will contain some interesting shadows.

3.7.3 Three-Point Lighting

*Positioning the backlight*

Now we add a *third* light, called the “backlight”. This is positioned behind and a little above the model, and serves to accentuate the upper silhouette (particularly the head and shoulders), and make our model stand out from the background. I set its energy to 2.0, to increase the effect. This screenshot shows what should be a good position for this light (it is directly above the green line of the Y-axis). Put it too close to Suzanne, and her bald head will glow a little *too* brightly. :)

*I'm ready for my close-up, Mr DeMille.*

Now the render looks like this. Does the effect look familiar? You should have seen something like it in countless close-ups in film and TV, as well as portraits.

3.7.4 Other Lighting Setups

Of course, there are countless other ways to light a scene. The above ones are mainly intended for close-ups and portraits. But where there are multiple characters in a scene, or even no characters at all and just the scene, you may want to position multiple lights to draw attention to some parts or characters while playing down other parts.

3.7.5 See Also

- [Portrait Lighting Setups](#) — a good intro to the different lighting setups commonly used for professional portrait photography.

3.8 Faked Global Illumination with Blender internal

Getting uniform studio lighting with the blender internal renderer



This tutorial assumes you know the basics of the blender interface and how to add objects, toggle editmode, and scale objects.

Now lets start blendering!

3.8.1 Preamble

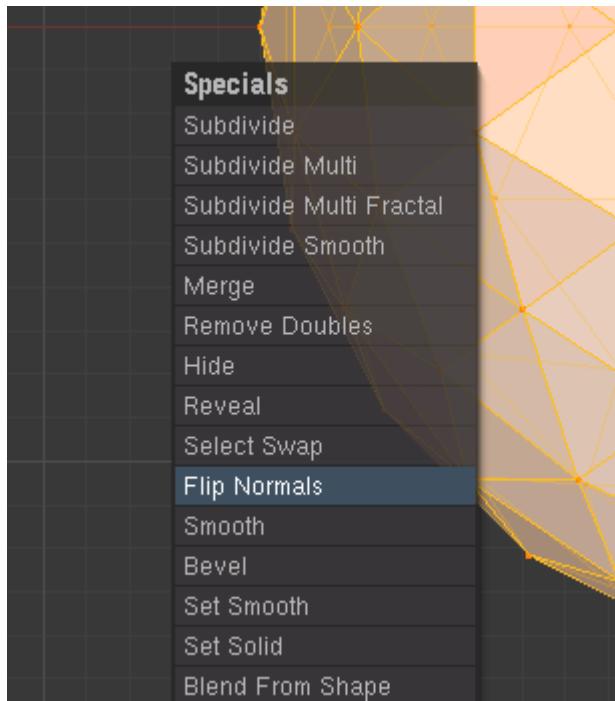
This tutorial will teach you how to use blender to create faked Global Illumination (here forth called “GI”). The reason it is faked is that blender currently doesn't support true Global Illumination but it's not a difficult task to fake. To fake global illumination, we will surround our subject with a lot of suns. The easiest way to do this is to create a very large sphere around our subject and place a sun at each vertex of this sphere, pointing inwards. Luckily, Blender can automatically duplicate an object at each vertex of another object. The results of this tutorial should look like this:

3.8.2 Blender Faked GI Tutorial

Add an ICO-sphere. Fire up blender and add an icosphere (**SPACE → add → mesh → icosphere**). Set the subdivisions to 3 and accept.

Scale the ICO-sphere by 15 times

Flip the normals of the icosphere so that they point inwards. This will make sure that our duplicated suns will point inwards. Go into edit-mode and press the A key till all faces/verts/edges are selected and press **WKEY** and *Flip Normals*.



Add a sun light source (SPACE → Lights → Sun)

Set the energy value of the light. This requires some special attention. If you keep the value at the default 1, you get a pure white, washed out scene because we will duplicate the light 162 times. A good way to calculate the light intensity required is to mess with the one sunlight and do test renders while tweaking the energy value of the sun as required till you get the brightness you want. THEN you divide the energy value of your sun by the amount of vertices in your icosphere and then finally set the sun energy to the number you got.

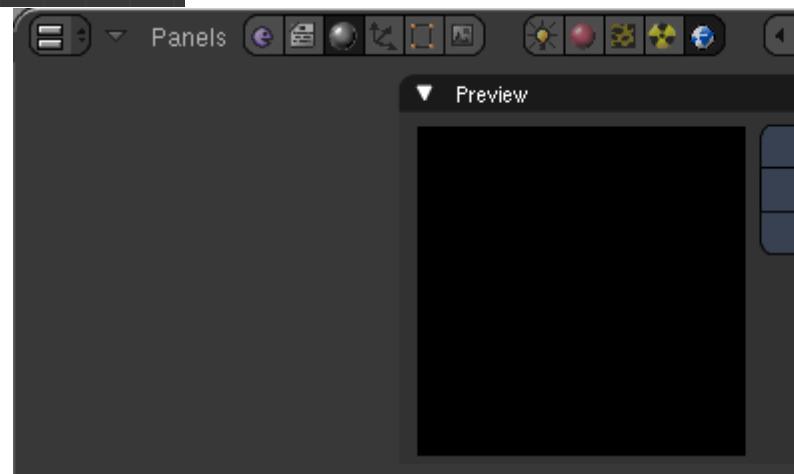
Noob Note: I found I had to set the energy of the sun lamp extremely low, at .044. Otherwise, you end up with a completely washed-out image.

Parent the sun to the sphere. Select the Sun THEN shift select the sphere and hit **CTRL+PKEY**

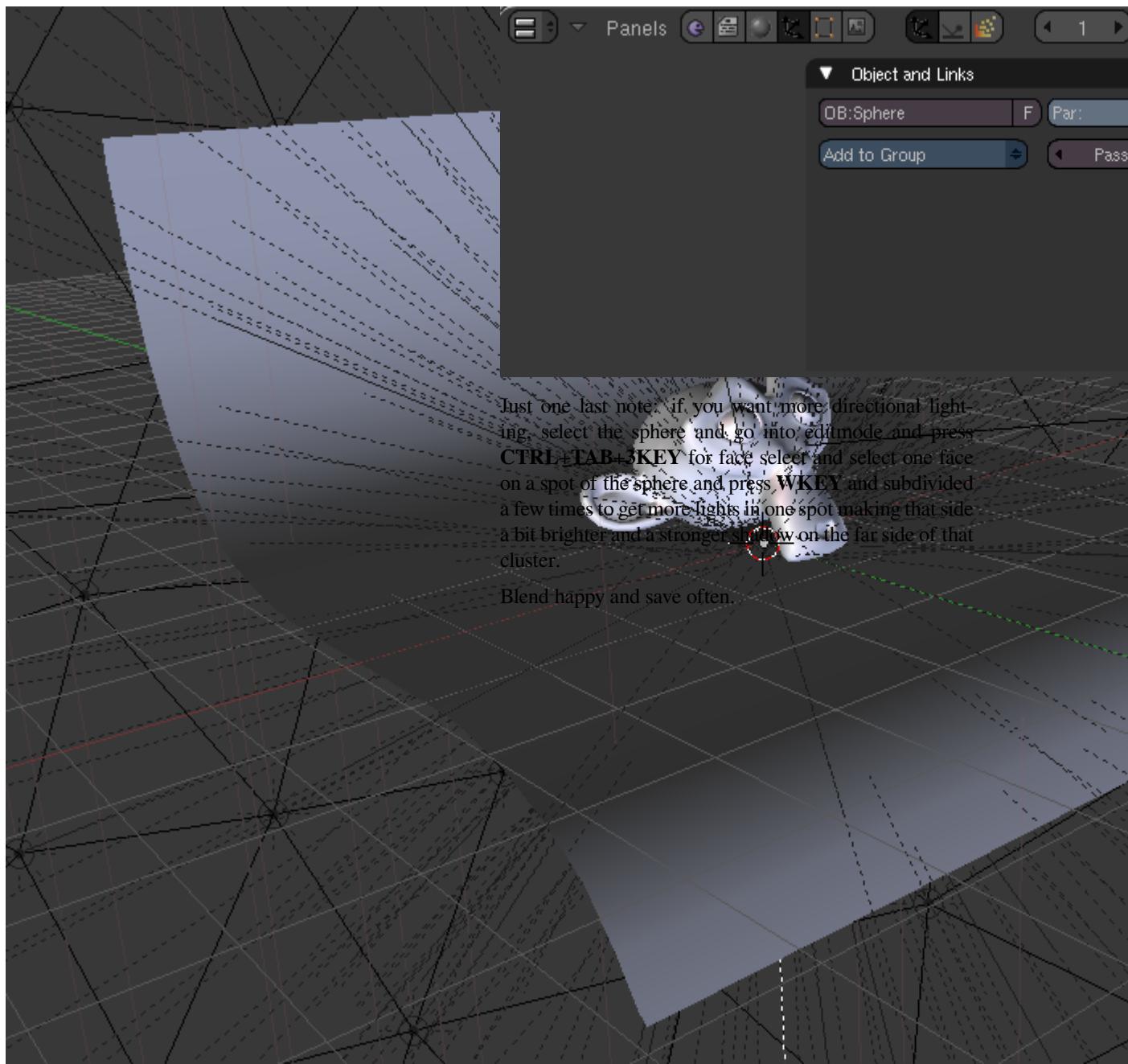
Enable dupliverts for the sphere. Depress the DupliVerts button. This will copy the sun to each vertex of the sphere. Also depress the ROT button. The ROT option tells Blender to rotate the suns to point along the normal of each vertex.

Depress the DupliVerts and the Rot but

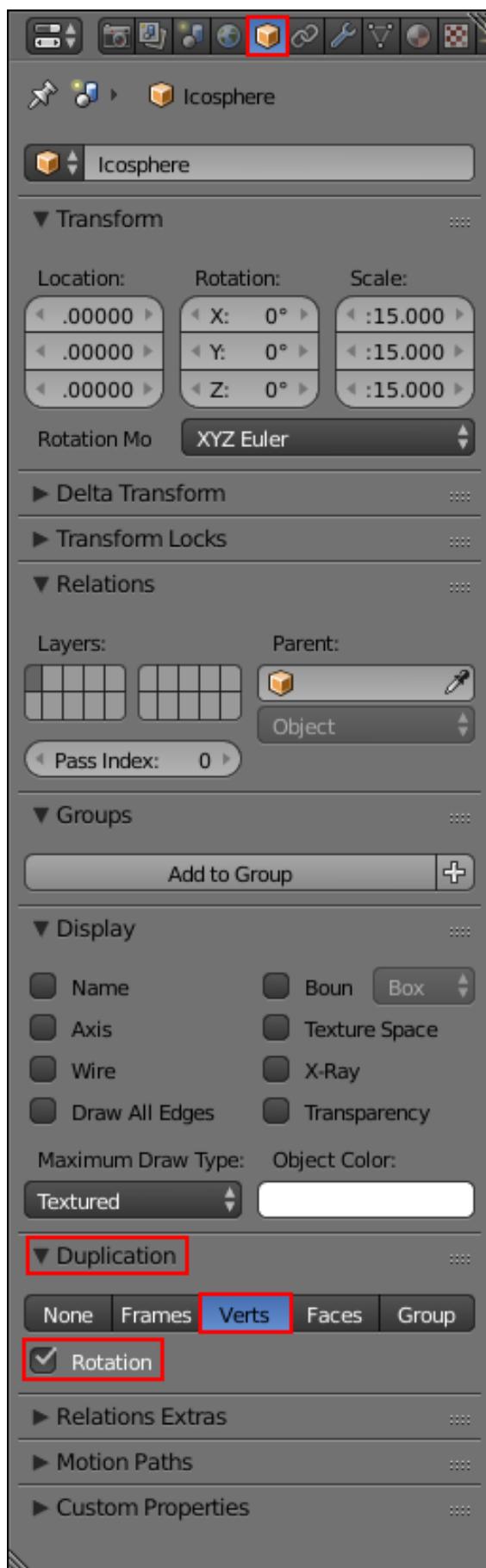
Enable AAO (Approximate Ambient Occlusion) This will result in more realistic soft shadows around our subject. Set the AAO settings in the world buttons as in the following image:



Set up the scene to render This can be anything you want but I used a simple curved plane with Suzanne on top of it.



One more note: Often it is hard to see your scene when you have this huge sphere encompassing your scene. So what you can do is to set the draw of the sphere to wireframe.



In more recent Blender versions *Dupli Verts* and *Rotation* options can be found under the *Object Data* tab in the *Properties Panel*. *Icosphere* needs to be selected.

3.9 Practising Good Parenting

In 2D drawing programs, you may be familiar with the concept of *grouping* objects together, so that they can be manipulated and transformed as an indivisible whole.

Blender can achieve a roughly similar effect through its concept of parenting one object to another. (Blender also has a concept called “grouping”, but that serves an entirely different purpose, which will not be discussed here, so don’t be confused.)

3.9.1 Parenting and Unparenting



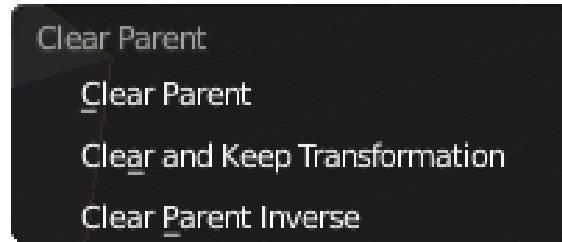
To parent one object to another is simple: in object mode, select the to-be-child object(s), then select the to-be-parent object, then press CTRL + P to bring up the “Set Parent To” menu. The options in the menu have the following meanings:

- Object: set the parent of the rest of the selection to the active (last-selected) object, clearing any existing parent relationships the new-children-to-be may have had. Any objects that were previously parented are reset to their original transformations before that previous parenting.
- Object (Keep Transform): similar to the above, except the children keep the transformations from any previous parenting they may have had.
- Vertex: this allows you to parent a child to a single, currently-selected vertex in the parent (which must be a mesh). You can go into Edit mode on the parent mesh to make the selection before using this option. The child will then track the movements of just that vertex, not the entire object.
- Vertex (triangle): here 3 vertices are selected in the parent mesh. The child tracks, not just their translations, but also their relative rotations as well.

When viewing your scene in Object mode, you will see black dashed lines connecting child objects to their parents. Try moving, rotating or scaling just the parent object, and you can see how the same transformation is automatically applied to its children as well.

To parent multiple children at once to a common parent, select all the child objects in turn, then *last of all* select the parent object. Now when you do CTRL + P , all the objects selected (except the last one) become children of the last one.

A parent of objects may in turn be a child of yet another object; any transformation of its parent will automatically be passed on to it and all its children, and their children, and so on.



To remove a parent relationship: select the child in Object mode, and press ALT + P to bring up the clear-parent menu. The options here are:

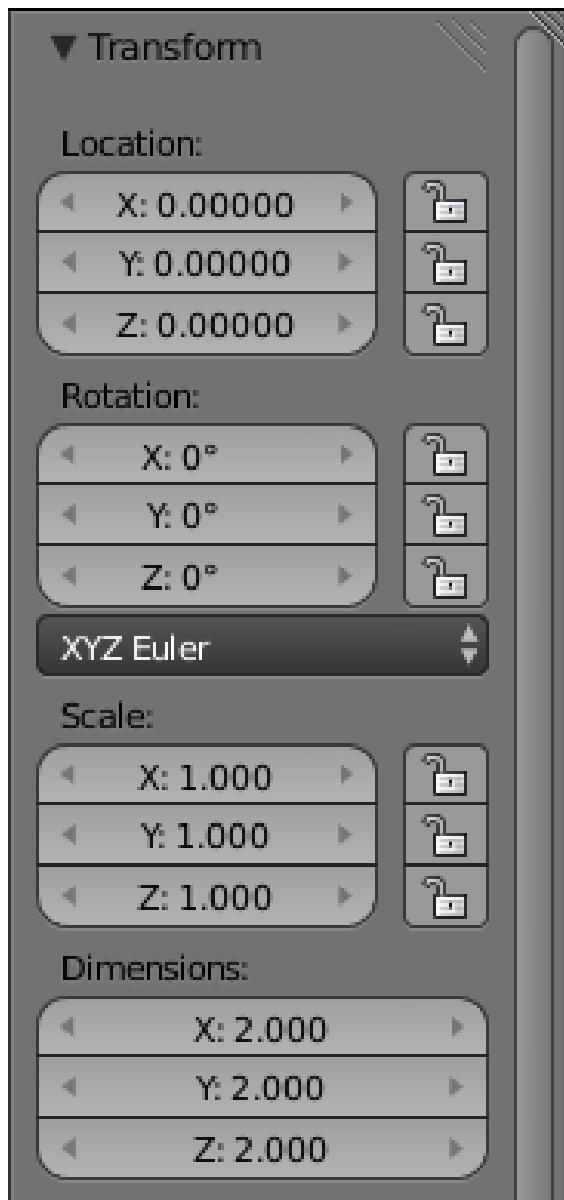
- Clear Parent: remove the parent/child relationship, and reset the child to the original transformation it had before the parenting.
- Clear and Keep Transformation: remove the parent/child relationship, but the child keeps its current transformation as a result of the parenting.
- Clear Parent Inverse: this doesn’t actually clear the parent/child relationship, but it sets the transformation of the child relative to the parent to the same as the transformation the child had on its own before it was parented.

Clear Parent Inverse Clarified

Normally, when a parent relationship is set up, if the parent has already had an object transformation applied, the child does not immediately inherit that. Instead, it only picks up *subsequent* changes to the parent’s object transformation. What happens is that, at the time the parent relationship is set up, the *inverse* of the current parent object transformation is calculated and henceforth applied before passing the parent transformation onto the child. This cancels out the initial transformation, leaving the child where it is to start with. This inverse is not recomputed when the parent object is subsequently moved or subject to other object transformations, so the child follows along thereafter.

The “Clear Parent Inverse” function sets this inverse transformation to the identity transformation, so the child picks up the full parent object transformation.

3.9.2 Lock Up Your Children!



In the Properties shelf at the right of the 3D view (you can toggle its visibility with N), you will see at the top the Transform panel. This shows the overall object transformation (translation, rotation, scaling), but note also the padlock icons next to the transformation fields: clicking each one closes its padlock, locking the corresponding field against further changes, *including changes made with the usual object-transformation tools in the 3D view*. (If you click a closed padlock, it will open again and remove the lock.)

This can be useful for a child object: if its object transforms are locked, it will still follow changes to transformations on its parent, but it cannot have its transformation changed directly. This can prevent accidents when manipulating an “object” which is actually made up of multiple Blender objects: parent them all to a common

root object (e.g. an Empty), and then lock them all, apart from the root object, and so they will transform together by manipulating just the root object, and cannot (accidentally) be separated.

For example, you might construct a car with separate objects for the doors and wheels; these might be left free to rotate (about their hinges and axles respectively), but are otherwise locked in position relative to the car body.

3.9.3 Example: Camera Pan

Imagine you want to make a movie where the camera does a 360° pan right around an object. We'll leave the details of how to set up an animation for later, but for now let's just consider how to set up the camera movement.

Start with a new default Blender document. With the 3D cursor at its default location (at the centre), add a new Empty object. It should end up inside the cube, so it won't be visible; switch Z to wireframe view, and it should be easier to see.

Select RMB the camera; then select SHIFT + RMB the Empty as well; now CTRL + P parent the camera to the Empty.

Now select RMB just the Empty, and try R otating it: notice how the camera follows along? For added fun, switch to camera view NUM0 (make sure the Empty is still the only thing selected). Now try rotating the Empty: it should look like the cube is rotating in the opposite direction, when it is really the camera moving around it.

3.10 Overview

Frames and Keyframes

A frame is a snapshot of the scene at one moment in time. An animation consists of displaying a succession of frames representing successive moments in time; if these are shown sufficiently quickly (at least 24 frames per second), the eye is fooled into seeing smooth movement, instead of a succession of still poses.

This is the principle behind both cinema film and digital video. But long before these were invented, it was known that you could make a sequence of drawings on pages of a *flipbook*, which could then be rapidly flipped by hand to produce an animation.

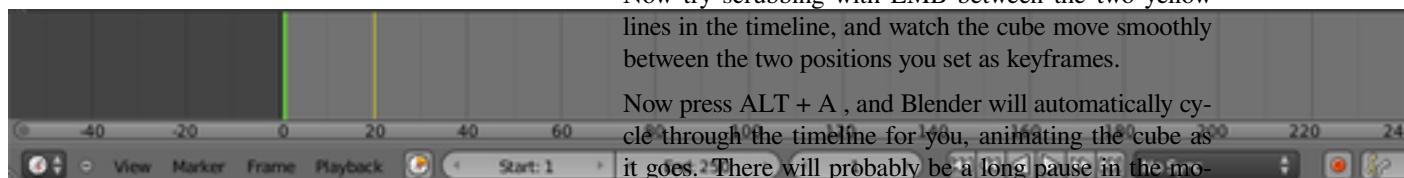
In live action video, we can capture the frames simply by letting the camera record as the scene unfolds. In hand-drawn animation (cartoons), each frame had to be drawn by a human animator (though there were some shortcut techniques like articulated character pieces, separately-moving scenery layers etc). Actually what would happen was that the most skilled artists would create keyframes representing pivotal points in the animation (starting and

ending poses in a character's movement etc), and the lower-paid assistants would have the job of filling in all the intermediate frames to produce smooth movement between those endpoints.

Computer animation works in a similar way, except here Blender is your lower-paid assistant. You go to crucial points in the timeline of your animation, position and pose your objects/characters appropriately, and tell Blender that this is a keyframe for the relevant transformations (positioning/rotation/scaling) of those objects/characters. Then when you run the animation, Blender will *interpolate* the specified transformation parameters between keyframes, giving you smooth motion over those intervals.

3.10.1 The Timeline

At the bottom of the default Blender screen layout is a window called the timeline . This gives you an overview of your animation.



You can zoom the view in and out with the mouse wheel, or scroll left and right with MMB .

The numbers across the bottom are frame numbers, with your animation starting at frame 1. The light grey background indicates the total duration of the animation. The vertical green line is positioned at the current frame time, and the current frame number is also displayed in the box between the start/end values and the transport controls, and at the lower left of the viewport in the 3D view window. Yellow lines indicate where keyframes have been inserted.

You can set the current frame time by clicking with LMB at the desired position. You can hop forward and backward a frame at a time with the left- and right-arrow keys, skip to the next or previous keyframe with the up- or down-arrow keys, and jump immediately to the first or last frame by holding down SHIFT and pressing left- or right-arrow.

You can also “scrub” by dragging with LMB across the timeline, which causes the animation to run backwards or forwards at whatever speed you choose, locked to the times across which you drag.

3.10.2 See Also

- The [animation section](#) in the Blender user manual.

3.11 Introduction to Keyframing

In this module, you will learn the basics of how to insert and remove keyframes, and preview the resulting animation in the 3D view.

3.11.1 First Keyframes

Start by opening a new Blender document. Select the default cube. Ensure that the timeline  is showing that you are at frame number 1. Press I , and choose “Location”; this will insert a keyframe at frame 1 which remembers the current location of the cube. Move the current frame (green line) away from frame 1, and you will see that there is a yellow line left behind.

Now go to, say, frame 25. With the cube still selected, press G , and move the cube to a different position—anywhere a few cube widths away will do fine. Press I again, and insert another Location keyframe.

Now try scrubbing with LMB between the two yellow lines in the timeline, and watch the cube move smoothly between the two positions you set as keyframes.

Now press ALT + A , and Blender will automatically cycle through the timeline for you, animating the cube as it goes. There will probably be a long pause in the motion after it gets to frame 25, because by default the animation will run until frame 250. Press ESC to stop the animation, click in the box at the bottom of the timeline labelled “End:”, and reduce the end frame number to, say, 50. Press ALT + A to start the animation again, and watch the movement cycle through a little more quickly this time.

Stop the animation, go back to frame 1, and press ALT + I to delete that first keyframe. Move the current time away from frame 1, and confirm that the yellow line that was there has gone. Now restart the animation; what happens? You should see the cube snap to the location specified by the only remaining keyframe, and stay there. Without a second keyframe specifying a different value for a parameter, Blender sees no reason to change the value for that parameter to anything else. Hence the rule:

3.11.2 Preview From All Angles

Press ESC to ensure the animation is stopped, then CTRL + Z to undo your deletion of the first keyframe above. Press ALT + A to start the cube moving between its two positions again. While it runs, the 3D View window remains fully operational; try using MMB to rotate the view, the mouse wheel to zoom in and out, NUM0 to toggle in and out of camera view, etc. All the while, the cube keeps running through the dance you choreographed for it. How neat is that?

3.11.3 What Is Being Animated?

Those yellow lines in the timeline aren't really very informative. They tell you there is a keyframe at that time, but not what settings are being specified. And there is no way to adjust an already-inserted keyframe, you have to delete it and insert another one.

All this fine control (and much more) is possible elsewhere, in the Graph Editor window, but we'll leave that for later. For now, just bring up the Properties Shelf in the 3D View by pressing N (if it's not already visible). At the top of this is the "Transform" panel, where you can see location/rotation/scaling settings for the currently-selected object. Notice that the Location values for the cube are displayed against a coloured background, either green or yellow; move the current frame time to a keyframe time, and it will be yellow, otherwise it will be green.

This coloured background is your cue that the specified value is being animated, and whether the current animation time is a keyframe time for that value. This goes for animating other properties as well, not just object transformations.

3.11.4 Animating a Material Property

Think of this next example as a teaser. It will be left incomplete for now, because to see the full effect you will need to put a bit more work into material, lighting and render settings, and render out the full sequence for viewing in some kind of external movie player. Feel free to come back and fill in the gaps as you learn more about those things.

One of the goals in the Blender 2.5x rework was to introduce the concept of being able to "animate anything". Animation had been a bit of an afterthought in earlier versions of Blender, but in 2.5x it is integrated very deeply, to the point where just about any object property can be animated over time.

Start a new Blender document. Select the default cube, and create a new material for it. You don't need to change the colour or any of the other settings for it, but look in the material settings for the "Transparency" panel and check the title box.

Look in that panel for the "Alpha:" slider (which might be abbreviated to something like "Alp", depending on the size of your screen). It should be showing the default value of 1.000 (fully opaque). Right-click on it (RMB) to bring up a menu with a bunch of options, of which the one we want is the top one, "Insert Keyframe". Once you select this, the background of the Alpha slider should turn yellow.

Set the current frame to another point, say frame 25. As you move away from frame 1, the background of the Alpha slider turns green, indicating it is being animated, but is not currently at a keyframe. At frame 25, set the Alpha

value to zero, or fully invisible (by either dragging across it with LMB or clicking in it and typing the new value). Then right-click on it and insert another keyframe.

Now try scrubbing back and forth in the timeline between frame 1 and frame 25, and you should see the Alpha value change accordingly between the two key values you specified. If you look in the preview image at the top of the material properties window, you should see the sample object there correspondingly fade in and out. Unfortunately the actual cube in the 3D view won't do this ("Solid" Viewport), but it will when you set the Viewport to "Material". If you were to render the image at the current frame with F12 , you should see the cube appear with varying degrees of faintness, down to becoming completely invisible at the end.

In short, you now have an animated disappearing cube.

3.12 The Ways of the Animator

You have just seen how animation can be applied to most object property values. In addition to this, Blender provides some specific features to aid in animating movements of parts of objects:

- Meshes can have *shape keys* defined for them. These give different positions to the vertices (though their number and *topology*—edge/face connections—cannot change). The amount of influence each shape key contributes to the shape can be continuously adjusted from nothing to 100%, and like other property values can be made to vary over time.
- *Armatures* are specialized objects, consisting of rigid *bones* that can be connected by joints, and moved and rotated relative to each other to produce different poses, very much like the skeleton of a human or animal. Armatures do not appear in the final render, but a mesh can be *deformed* (i.e. have its shape changed) by an armature to bring lifelike movement to a character.
- *Lattices* are another kind of specialized object that can also be used to deform a mesh. Like armatures, they do not appear in the final render. Unlike armatures, they lack rigid joints, and so produce more rubbery changes of shape, characteristic of creatures without bones, or perhaps for a cartoony effect.
- Curve and surface objects can also have shape keys defined for them. And of course curves can be used to deform a mesh via the curve modifier.
- You previously saw the usefulness of the **Empty object** in modifiers. But being an object with position, scale and rotation properties like any other, these properties can be animated for an Empty, too, with

corresponding effects on those modifiers. Also if the Empty happens to be a parent of other objects, then those objects can be animated as a group, just by animating the Empty.

3.13 Animation Editors

Blender offers several different kinds of editing windows specifically to do with animation.

3.13.1 Timeline

You have previously come across the Timeline  view as the simplest way to see the keyframes in an animation. This window also includes transport controls that you can use to start and stop the animation, jump to a particular frame with LMB , and scrub by dragging across a time range with LMB .

This window also lets you define *markers* which you can name to identify important points in the animation, for informational purposes. Also, using the “Bind Camera to Marker” function, you can dynamically switch the active camera at marked points in the animation, to cut between different viewpoints.

3.13.2 Graph Editor

You previously saw how to define keyframes for animating a property, such as an object position or material setting. But what if you get a keyframe definition slightly wrong? Perhaps the timing is slightly off, or the movement is not quite right. You could delete the keyframe and try again. But, in many cases, it would be easier if you could get into the actual definition of the keyframe and tweak it around a bit, move its time position, adjust the animated property values, that kind of thing.

This is the purpose of the Graph Editor . It gives you the most detailed, low-level view possible of your animation, by drawing an *FCurve* for each animated property of the selected object, representing how the property value varies over time. Each curve has control points located at each of its keyframes, and you can move these points around—horizontally to change the timing, or vertically to change the keyframe value—as well as add and remove points.

3.13.3 Dope Sheet

The Dope Sheet  gives a high-level view of your animation, similar to the Timeline, but slightly more detailed. Here you see the separate keyframes for each animated object, and you can do some limited editing, like

moving the keyframes around, and duplicating and deleting keyframes, but you can't seem to add entirely new ones.

Actually, this window can show any of five different editor submodes: Dope Sheet, Action Editor, Shape Key Editor, Grease Pencil or Mask. The Action Editor is useful in conjunction with the NLA Editor (below).

- See also: [Dopesheet](#) on the Blender Wiki.

3.13.4 NLA Editor

The NLA (“Non-Linear Animation”) Editor  represents a different way of setting up an animation: instead of thinking of it as a single linear sequence of keyframes, you break it up into *actions*, which are separate sequences that can be arranged in various ways. You can also duplicate an action any number of times, position the copies at different times in the overall animation, and even attach them to different objects. But they still share a single set of FCurve contents, which you can edit in the Graph Editor as you would a monolithic linear animation sequence, and your changes will take effect in all copies of the action.

To edit the contents of an action in the Graph Editor, you select the action and TAB into “tweak mode” (analogous to Edit mode in the 3D view). TAB again takes you out of Tweak mode.

However, the Action Editor submode of the Dope Sheet now becomes useful, for creating new actions. These start out initially empty of any keyframes or FCurves, but you can add these in Tweak mode.

- See Also: [NLA Editor](#) on the Blender Wiki.

3.13.5 To Summarize...

The above description may seem rather confusing in places. Basically, there are two ways to organize your animation:

- Linear, as a single sequence of FCurves and keyframes
- Non-linear, as a sequence of actions, each consisting of a sequence of FCurves and keyframes. The nonlinear data can be further divided into multiple overlapping *tracks*, which combine together at any moment in time to produce the complete animation.

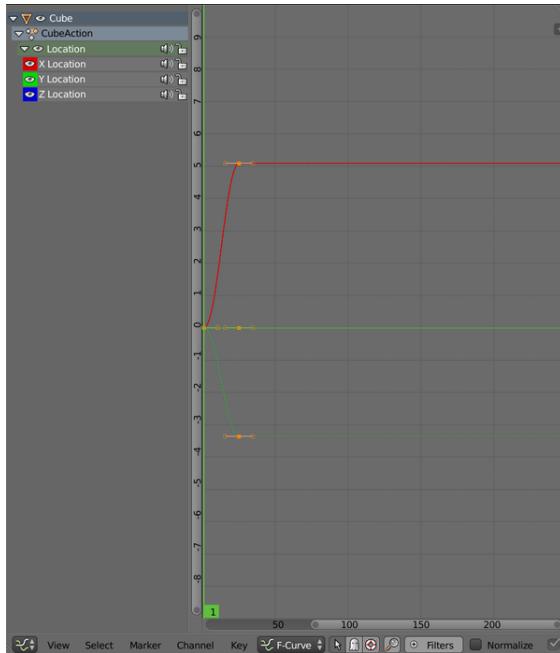
Objects that can have animation data attached to them can have both kinds, linear and nonlinear. You can even attach both kinds at once, though then the linear animation data takes precedence. It is easy to change your mind and switch back and forth; typically, the development of

a simple animation might start out linear, then change to nonlinear as it gets more complicated.

3.14 Introducing the Graph Editor

The Graph Editor is where you exercise the lowest-level control over the details of an animation, right down to the settings and placement of each keyframe. You previously learned how to create and delete keyframes; now you will learn how to get inside them and tweak them.

Let's start by creating a very basic animation, using the default cube. As you did in "First Keyframes", set up a couple of Location keyframes, one at frame 1, and another positioning the cube a few widths away at frame 25. Scrub between these times in the Timeline, and confirm that the cube moves between those two positions.



Now, add a new window to the left of the 3D view and change its type to Graph Editor . Provided the cube is still selected, it should show something like this. The Location controls at the left will likely initially appear collapsed to a single line; click on the right-pointing triangle so it points downwards, revealing the separate X, Y and Z Location control lines.

The curves at the right are the *FCurves*; the colours for the X, Y and Z location curves correspond to the colours of the squares surrounding the eye icons in the control lines at left.

Note the following features of the display:

- The vertical green line represents the current frame. It always shows the same frame as the correspond-

ing line in the Timeline , and also the frame number at the lower-left corner of the 3D view . Just as in the Timeline, you can set its position with LMB , or use the arrow keys to step forward/backward one frame (right/left arrows) or jump to the next/previous keyframe (up/down arrows).

- There is also a horizontal green line which is positioned with LMB . This can be used (together with the vertical line) for snapping selected control points to the specified value or time.
- The control points on each curve, at the position of each keyframe. These are black when not selected, or a pinkish colour when selected. Various keyboard shortcuts you should be familiar with from the 3D view also apply here: RMB to select one point (deselecting everything else), SHIFT + RMB to add/remove a point to/from the selection, A to toggle selecting everything/nothing, even B to do a box selection, even C ircle select is available.
- The little handles joined to each control point by straight lines. FCurves are *Bézier curves*, just like those commonly found in 2D illustration programs, or in Blender's own **curve objects**. By default each control point gives a smooth curve, causing the position (or other object property) to smoothly accelerate and decelerate; but you can change the handle type with V , if for example you want a more sudden change.
- The eye, speaker and padlock icons in each control line.
 - Clicking the eye icon causes the eye to close, and the corresponding FCurve to disappear from the display; this reduces clutter and makes it easier to edit just the right curve. Click the closed eye to open it again and make the hidden curve reappear.
 - Clicking the speaker icon *mutes* (temporarily disables) the effect of the corresponding curve, without actually deleting it. This may be useful for debugging complex motions. When muted, the radiating arcs disappear from the speaker, and the curve turns white; click the speaker again to unmute the curve.
 - Clicking the padlock locks the curve against selection and editing, while keeping it visible. Clicking on the closed padlock opens it again, restoring the ability to select and edit the curve.

- Clicking on the text or background (i.e. not in any icon) on a control line with LMB selects the entire curve. You can also SHIFT + LMB to select multiple curves at once. Pressing DEL or X within the area of the control lines will get rid of the **entire** selected curve(s)!

- You can scroll around the view with MMB . Or you can drag the scroll indicators, anywhere except at their ends.
- The view can be zoomed with the mouse wheel, as in the 3D view. However, this zooms both horizontally and vertically by the same factor. It is characteristic of the Graph Editor that the curves are likely to be stretched very tall, while squashed very narrow. To zoom on each axis independently, drag on the ends of the scroll indicators: lengthen them to zoom out, shorten them to zoom in.
- You can also use Home and NUM. , just like in the 3D view, to zoom and centre the view on the entire set of curves or just selected curves or parts thereof.

Set the current frame to something in-between the two keyframes, e.g. frame 12. Select RMB one of the control points at frame 25, and try moving G it around: how does it affect the cube in the 3D view? Constrained moves are very useful here:

- G Y to change the value of the FCurve at the control point without moving the keyframe in time
- G X to move the control point in time without changing the value it gives to the FCurve. You can move one control point past another; Blender will automatically reconnect the curve segments to ensure the curve does not loop back on itself.

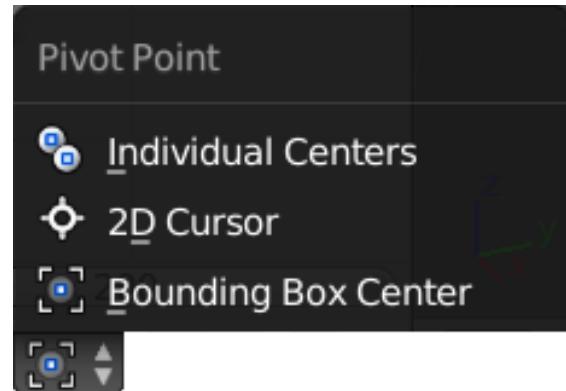
Also, go to frame 25 (the time of the second keyframe), and make sure nothing is selected. Now go to the Select menu, and choose the option “Column on Current Frame” (keyboard shortcut CTRL + K): this selects all keyframes located at the current frame time. Now you could, e.g. use G X to adjust the position of all control points for this keyframe in time.

Inserting and removing control points: You can, of course, delete selected control points with DEL or X in the usual way. You can also insert control points at the current time with I : this will pop up a menu asking whether you want to insert control points for all editable channels (FCurves), or only the curves containing currently-selected control points. The new control points will be inserted so as to make minimal difference to the actual shapes of the curves; but you can of course tweak them around afterwards.

Rescaling an entire animation sequence: Supposing you painstakingly set up an animation sequence, only to discover that the total length of it is not quite right, either too long or too short. It is easy enough to lengthen the entire sequence (making it run slower), or shorten it (making it run faster).

First of all, make sure all the affected objects are selected. In the Graph Editor, ensure all the relevant FCurves are visible and not locked. Set the current time to frame 1.

Press A once or twice to ensure that *all* control points are selected.



In the window header for the Graph Editor, find the Pivot Point menu (as at right). Make sure the “2D Cursor” option is selected. Now press S X to rescale the entire animation in time, keeping only keyframes at the current time (which you set to frame 1, remember) unchanged. Either type in a suitable number to get an exact new length, or adjust it by eye so the last keyframe ends up at the right frame number. As always, press ENTER to confirm the transformation.

3.15 Animation Rendering

It’s time to revisit the rendering settings we looked at when we were first learning about rendering single still images, and go over some of the stuff we skipped back then.

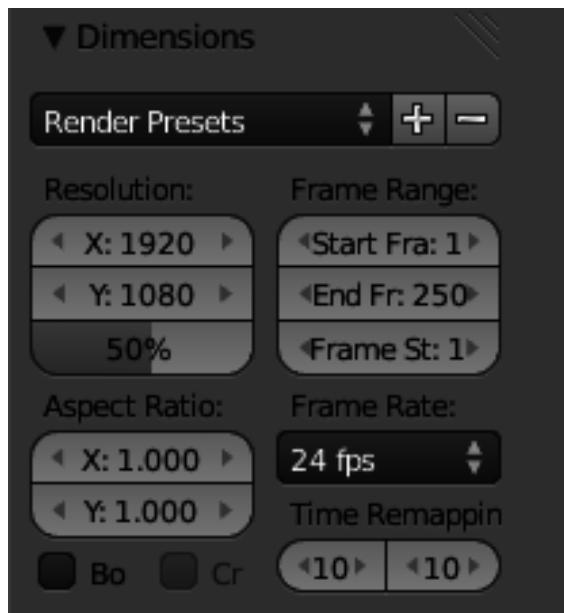
3.15.1 Render Control



We previously skipped over the second and third buttons in this panel: the second button is equivalent to CTRL + F12 (render animation), while the third one invokes your player, as configured in your user preferences, to play a rendered animation.

3.15.2 Image Dimensions

We previously looked at the settings in the left column (spatial dimensions of the images), now look at the ones



on the right (time dimensions).

The “Start Frame” and “End Frame” numbers simply mirror the same values at the bottom of the Timeline window; changing these numbers in either place automatically changes them in the other place as well. These numbers determine which part of the animation sequence you want to render. These numbers are *remapped* frame numbers (see below about time remapping), so they don’t necessarily correspond directly to frame numbers in your timeline.

The “Frame Step” number, if set to a value n greater than 1, tells Blender not to render every frame, but only every n th frame. Like the scale factor in the image resolution, this gives you a faster render, handy for previewing, at the expense of lower quality.

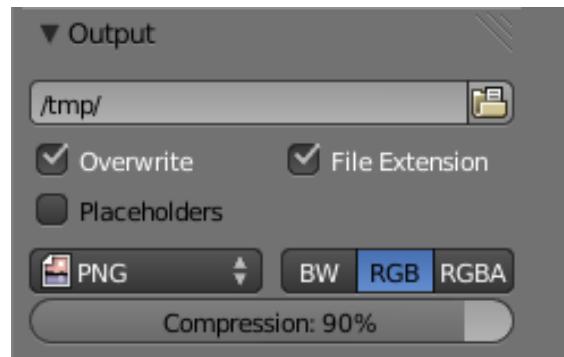
The “Frame Rate” controls how timeline frame numbers relate to actual time. The default value of 24fps is the usual value for film, while 25fps is the standard value for PAL video and 29.97fps for NTSC.

The “Aspect Ratio” provides a pair of divisors for scaling to account for *nonuniform* pixel densities in the X and Y direction. If you are creating footage for the old standard-definition PAL or NTSC video formats, then these need to be set to suitable values (which you don’t have to calculate, since they can automatically be filled in by choosing the appropriate item from the preset menu). Otherwise, leave them at the default 1.0/1.0.

The two fields labelled “Time Remapping” let you warp time, and speed up or slow down the animation from its original rate. The “old” value (on the left) is converted to a time interval in seconds by dividing by the specified frame rate, and then the “new” value (on the right) determines how many frames are actually rendered during that interval. If both values are the same (the default), then

time remains unwarped, and the frame numbers correspond directly to timeline frame numbers.

3.15.3 Image File Formats



Blender’s output file formats include regular movie formats (e.g. MPEG, Ogg Theora), but you can also use a still-image format (e.g. PNG, OpenEXR). Outputting an animation to a still-image format involves generating a separate file for each frame, named according to the frame number, into the directory that is named in the box at the top of the “Output” panel.

This editable field gives the location for saving animation renderings. Only animation sequences are automatically saved here; single still frames have to be explicitly saved from the Image Editor. If the path begins with “//”, then it is interpreted as a subdirectory of the parent directory containing the .blend file.

Outputting straight to a movie format is fine if you don’t intend to do further processing of the resulting movie. Trouble is, that’s rarely true. You might need to adjust the images in some way, or edit the rendered sequence into a longer production along with other pieces. But most of these movie formats are designed only for playback, not for such additional processing and editing, and such processing requires decoding the movie format, doing the processing, then re-encoding again. This is generally a time-consuming process, and it leads to loss of quality.

And you might discover that your renders didn’t come out right in the first place, and so you need to redo it all. It is quicker to get to this stage if Blender doesn’t have to do movie encoding as well.

Therefore it is usually better to render your animation to a sequence of still frames. The files end up much larger, and there will be an awful lot of them, but they leave maximum room for you to do further processing with a minimum of quality loss. Once you have finished doing all your processing and editing, you can then encode the results to a regular movie format for delivery to your users, using a tool like FFmpeg which offers all kinds of fine-tuning over the encoding process that Blender cannot match. Nor is it Blender’s job to match.

Having said all that... In these tutorials, you probably want to generate renders quickly and conveniently, without worrying too much about quality. For that purpose, by all means choose one of the “Movie” formats for animation rendering.

If you don’t already have a player to hand that will handle any of the formats that Blender can generate, why not try VLC, which will play just about anything.

3.15.4 Single Frame Versus Entire Animation

You already learned how to use F12 to render a single still frame. In fact this renders the *current frame*—i.e. whatever the state of the model/scene is at the current frame time. Thus, you can go to different frame times and hit F12 to check how things look at crucial points, before doing a render of the entire animation, to save time in case you discover something doesn’t look right.

Once you are satisfied with the state of the entire animation, you can go to the “Render” menu and select “Render Animation” to render the whole thing (or hit **CTRL + F12**, unless your GUI has usurped this for another purpose). Of course, you might then discover further things that don’t look right, that you didn’t notice before, and have to go back and fix them before rendering again. This is why it’s a good idea to user lower-quality render settings to check how things are going from early on. Or in other words,

3.16 Lattice Modifier

3.16.1 What is a Lattice?

A Lattice is essentially a simple container that can be used to deform and manipulate a more complex mesh in a non-destructive manner (i.e. A lattice can be used to seriously deform a mesh then, if the lattice is later removed, the mesh can automatically return to its original shape).

3.16.2 How to add and use a Lattice

A Lattice is added to the scene in the same way other objects are added. Either:

Noob Note For Blender 2.62 or 2.63 users, I recommend referring to this [Lattice Modifier video tutorial](#) by Ed Lazor for animating the lattice, as numerous interface changes have occurred between Blender 2.4x and 2.6x.

1. Shift + A over the 3D window and choose **Lattice** from the pop-up menu, or
2. Press Space over the 3D window and choose **Lattice** from the pop-up menu (*Note: does not work in recent*

versions of Blender.), or

3. Click **Add** in the 3D View window header (located at the bottom of the 3D View) and choose **Lattice** from the drop-down menu

The default Lattice looks just like a cube when first added except that it is just one Blender Unit (BU) wide whereas a mesh cube is 2 BU wide. When the Lattice is added, the window remains in **Object Mode** and the Lattice can be moved, resized and rotated like any other Blender Object.

On its own, a Lattice serves no purpose whatsoever since it can’t be seen in a rendered image. Its only use is to manipulate another object and so, to be useful, we need to associate another object with it. The current method for doing this is by applying a “**Lattice Modifier**” to a Blender object.

The basic workflow is:

- Add a mesh (cube, cylinder, cone, sphere, etc.)
- Add a Lattice
- In the Object Context  of the Properties window, enter the name of the Lattice (The default name is Lattice, Lattice.001, Lattice.002, etc. - or you can give it a useful name that you'll remember later)
- Select the mesh
- Go to the Modifiers Context  in the Properties window. Click **Add Modifier** > Lattice

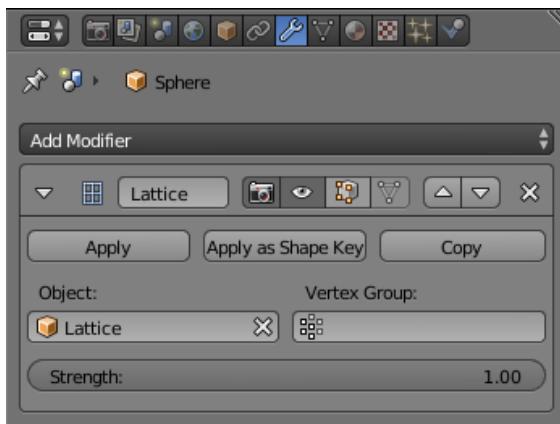
NOTE: *In older versions of Blender, a lattice was applied using a parent method - select the mesh, then select the Lattice, then press Ctrl + P and choose “Lattice Deform”. While this method is still available, it does not offer the full functionality of the newer Lattice Modifier. If you use the parent method, you will need to press the “Make Real” button in the modifiers palette to enable the modifier functionalities. Using the method, the Lattice becomes a parent and a modifier. As a parent, it will act on the mesh in Object Mode whereas a true modifier lattice only influences the mesh when altered in Edit Mode.*

Now, we can change the Lattice in **Edit Mode** and any changes we make to it will affect the mesh.

Note: Applying the Modifier Correctly When adding the modifier to your object, you must enter exactly the same name as your lattice. The default is **Lattice** with a capital “L” but if you’ve changed it or have more than one Lattice, then you will need to enter the new name. When you enter a valid Lattice name, the name stays visible in the Ob: box. If your entry disappears then you’ve entered the name incorrectly.



Adding a Lattice Modifier to an object



Lattice modifier interface in Blender 2.78. The modifier affects object Sphere. Effect is governed by lattice object named Lattice.

One simple way to get the right name is to select the Lattice, go to F9 , Link and Materials panel and where it says Ob:Lattice or Ob:Lattice.001 etc., move the mouse over this field and press Ctrl + C (*don't click on it, just hover over it*). This copies the Lattice name. Then select your object, go to the Lattice Modifier panel, hover the mouse over the Ob: field and press Ctrl + V to paste the name in. Now it should stay there and your Lattice should work in Edit Mode.

Noob note: I recommend the method with Ctrl + P for animation.

Basic Exercise:

- Start with a new Blender scene, delete the default cube and add a UVsphere (Shift + A > Mesh > UV sphere). Accept the default 32 Segments and Rings. You could use any of the mesh shapes but because the sphere is heavily subdivided (*made up of lots of edges and faces*), you will get a better idea of what the Lattice is doing.
- Tab back into Object Mode then add a Lattice. It is generally wise to resize the Lattice so that it surrounds the mesh it will be deforming. So, press S and enlarge the lattice to 2 BU wide.

3. Select (RMB) the Sphere and add a Lattice Modifier to it. Make sure that you type the exact Lattice name correctly into the box. Nothing appears to happen but that's fine. In newer Blender versions click the field under the label "Object:" and select the desired lattice object from the drop-down menu.

4. Now select your Lattice, Tab into Edit Mode, select just one control point (a control point looks like a vertex) on the Lattice and move it around. You will see the sphere mesh stretching and squashing relative to that control point. Move each Lattice control point one at a time and see just how far you can deform the mesh. Zoom in your 3D window if you need a closer look.

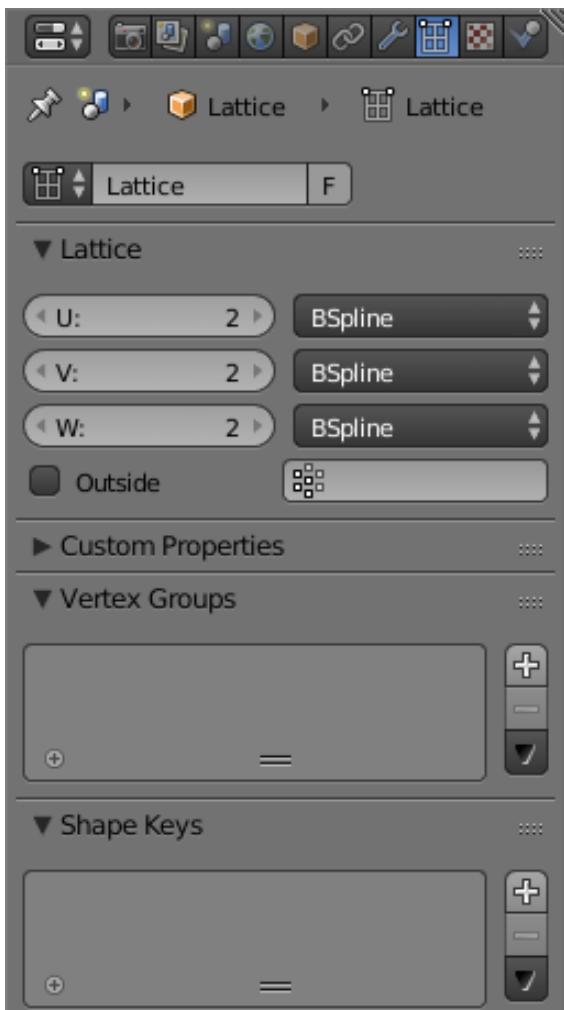
5. The control points of the Lattice can be moved, scaled and rotated in the usual ways. Try selecting a few of all the control points and scaling them (S) or Rotating them (R) and watch the mesh follow along.

6. Now, exit Edit Mode (Tab) and select the sphere in Object Mode. Go to the modifiers panel and press the big "X" button to remove the modifier. Despite all that deforming, the sphere immediately returns to its original size and shape. Nothing was really changed in the sphere's mesh data. Immediately re-apply the modifier as before and see the Lattice immediately apply its own deformation to the sphere again.

3.16.3 Getting more involved with Lattice

The default Lattice is two control points high, two wide and two deep, i.e., it is two control points wide in each direction (these are referred to as the U, V and W directions). However, we can change the number of control points in one, two or all directions. This is done by selecting the Lattice, going to the Lattice Panel (F9) and changing the values in the U, V & W buttons. If you decrease one figure to a value of 1, the Lattice will become two-dimensional (planar). Decreasing two values to 1 will change the Lattice to a line (one dimension). This can be useful, especially if the remaining value is increased, but is not the most common usage of Lattice.

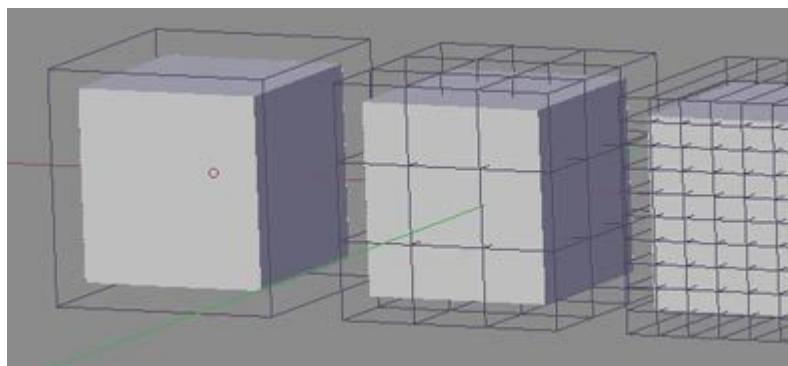




Lattice (Object Data) Context in the Properties window in Blender 2.78.

NOTE: The “**Make Regular**” button will set the UVW control points of an unscaled Lattice to be exactly one Blender Unit apart in each direction. The “**Outside**” button effectively removes all the internal control points which are added when the UVW settings are higher than 2.

In most situations you will want to increase some or all the values as this gives you more control over complex deformations, much like a subdivided mesh. How much you increase the UVW directions depends entirely on how much detailed control you need over deformations. As you've already seen basic squash, stretch, shear and simple deforms can easily be achieved with the default 2,2,2 Lattice but by increasing the UVW values a little, a whole new range of deform possibilities becomes available.

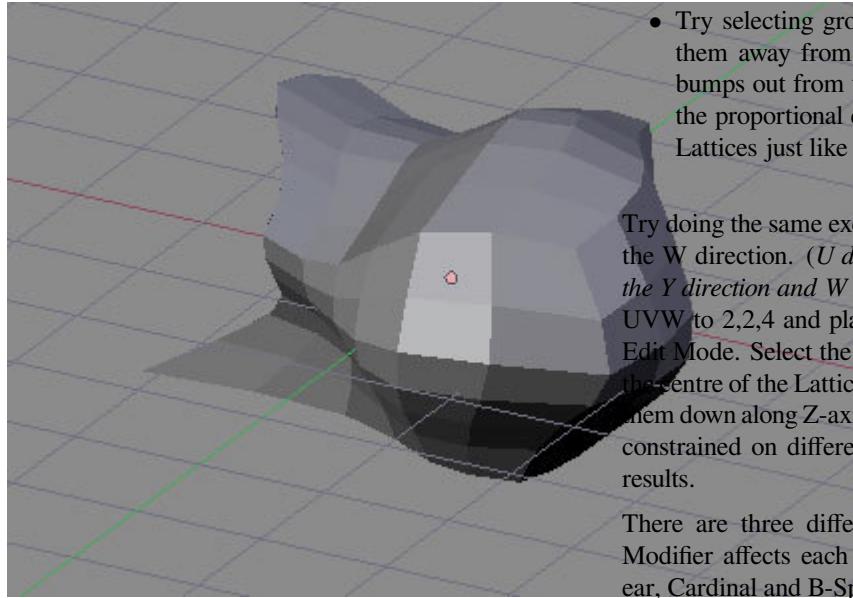


Three Lattices with different UVW settings

One thing to understand, however, is that a Lattice cannot bend the individual edges of a mesh (the lines connecting any two vertices) so the mesh must contain enough edges in order to apply complex lattice deformations to it. These edges must be genuine edges and not the virtual edges created by a subsurf modifier (Correction: Yes, they *can* be edges generated by a subsurf modifier, provided the subsurf comes above the lattice in the modifier stack for the object being deformed; use the up/down-arrow buttons next to each modifier to rearrange them as necessary). Edges can also be added to a mesh using a variety of tools including subdivide, knife and loop cuts (some info on these []).

New Note: Use K to open a menu for the knife, loop cuts, etc. Knives are used to add a vertex to each line you left click and drag the cursor over, and loop cuts allow you to make multiple cuts into your active object.

About loop cuts: Ctrl + R also is a direct shortcut to a loop cut. Furthermore, loop cuts' main uses are for subsurf modeling. In order to accomplish a hard edge in a subsurfed object (for example the edges and corners of a wooden table or the bumpers of a car) you must apply loop cuts to the faces surrounding the edge and place them very close to them (and parallel). Experiment with a simple subsurfed cube and you'll see. Keep in mind this is why when modeling it is highly recommended to use quads. A loop cut will only cut a quad face, or every face connected with it in one direction, but only quads. Another main reason why it is recommended to model with quad faces is regarding animation but I suppose other tutorials will deal with that later. More information:



A simple subdivided cube after a complex lattice deformation

- Try selecting groups of control points and pulling them away from the Lattice to stretch lumps and bumps out from the mesh. If you are familiar with the proportional edit tool, try that too - it works on Lattices just like it does on meshes.

Try doing the same exercise but increasing values in only the W direction. (*U divides in the X direction, V divides the Y direction and W divides the Z direction*). So, set the UVW to 2,2,4 and play with the control points again in Edit Mode. Select the two rows of control points around the centre of the Lattice and scale them up (S) then Scale them down along Z-axis only (SZ). Try other transforms constrained on different axes for interesting, controlled results.

There are three different options for how the Lattice Modifier affects each UVW direction. These are Linear, Cardinal and B-Spline. All I can say is that B-Spline is the default and to find out what the difference is, press the buttons and see.

Intermediate Exercise

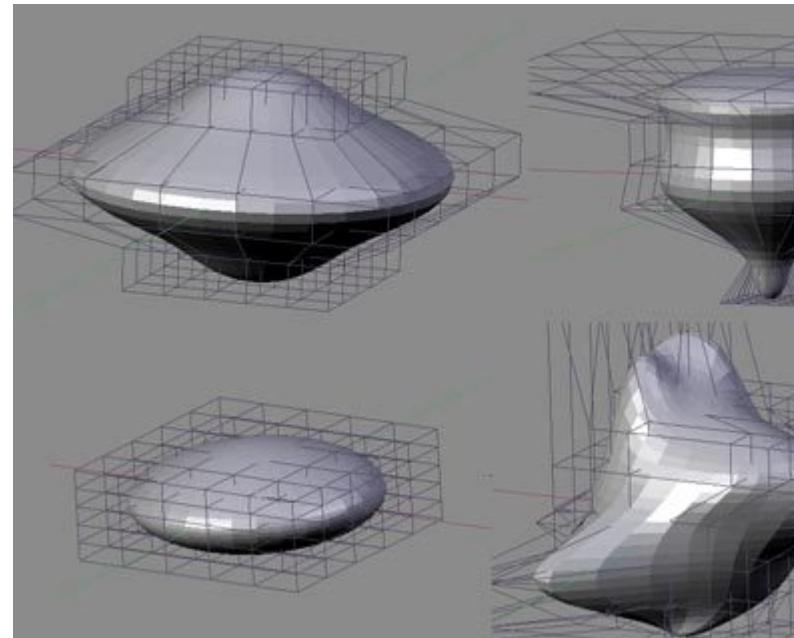
For this exercise, start over with a new scene and repeat Steps 1 - 3 in the Basic Exercise above. Don't go into Edit Mode yet.

Basic Exercise:

1. Start with a new Blender scene, delete the default cube and add a UVsphere (Spacebar>Add>Mesh>UVsphere). Accept the default 32 Segments and Rings. You could use any of the mesh shapes but because the sphere is heavily subdivided (made up of lots of edges and faces); you will get a better idea of what the Lattice is doing.
2. Tab back into Object Mode then add a Lattice. It is generally wise to resize the Lattice so that it surrounds the mesh it will be deforming. So, press S and enlarge the lattice to 2 BU wide.
3. Select (RMB) the Sphere and add a Lattice Modifier to it. Make sure that you type the exact Lattice name correctly into the box. Nothing appears to happen but that's fine.

- In Object Mode, select the Lattice then press F9 and go to the Lattice Panel. Increase all the UVW values to 4. You will immediately see the change displayed in the 3D window.
- Tab into Edit Mode and play with the control points again. The first thing you might notice is that the corner control points have less influence now than before. This is because the effect is proportional to the distance of each point from the mesh. Move some points in the middle of a side face and you'll see the effect is significantly greater.

3.16.4 Examples



Some examples of a Lattice deforming a standard UV Sphere mesh (not smoothed)

3.16.5 Making it stick

To keep the mesh deformed permanently, you can select it and press the “APPLY” button in the Lattice Modifier panel. This “bakes” the mesh in the deformed position and disconnects it from the Lattice. The Lattice can then be deleted without the mesh returning to its original shape. This is useful if you are using the Lattice as a modelling tool rather than an animating tool.

Also, instead of deleting the Lattice, you can use it to immediately modify another mesh. Simply move the Lattice over the new mesh in Object Mode then apply the Lattice Modifier to the new mesh. If the Lattice is already in a deformed state, the mesh will immediately be deformed too. Press Apply again to “bake” that mesh and keep re-using the already deformed Lattice on as many other meshes as you wish for matching results.

[Noob Note: I found it interesting that you can even apply the lattice deformation to the original object again once baked, and it will deform even further.]

3.16.6 Animating a Lattice

Thorough instructions for old Blender versions

Lattice animation uses a workflow almost identical to the RVK (Relative Vertex Key) workflow from Blender 2.37 and earlier. *You don't need to know this but some people might find the information useful.*

With your Lattice Modifier already added to your object, select the Lattice and press I to tell Blender you wish to animate the Lattice. Choose “**Lattice**” from the pop-up list. This sets the basis key (undeformed state) for the Lattice.

Adding the basis key

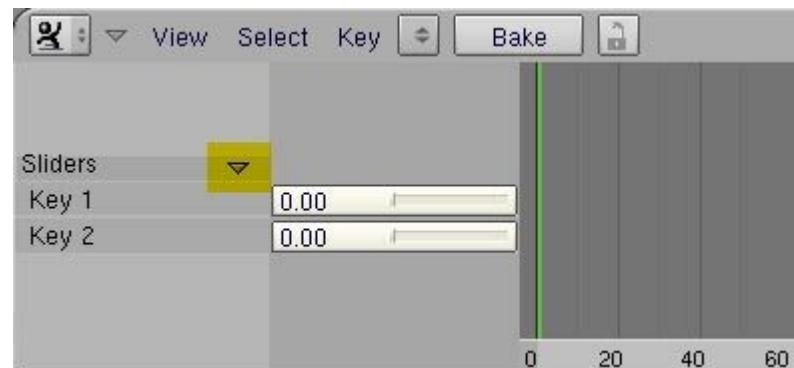
In the Editing panel (F9), press the “**Relative**” button.



Set the keys to “Relative”

NOTE: The Slurph setting determines the delay, in frames, for how long it will take to morph from their former state to the one applied by the new lattice.

To set your first deform key, press **I-Key** and then “Lattice” again then enter **Edit Mode**. Deform the Lattice by scaling or moving control points then Tab back into **Object Mode**. If you open an **Action Window** now, you will see your first key, “Key 1”, added to the list. If you press the small arrow at the top of the list, it will display the **slider** for that key.

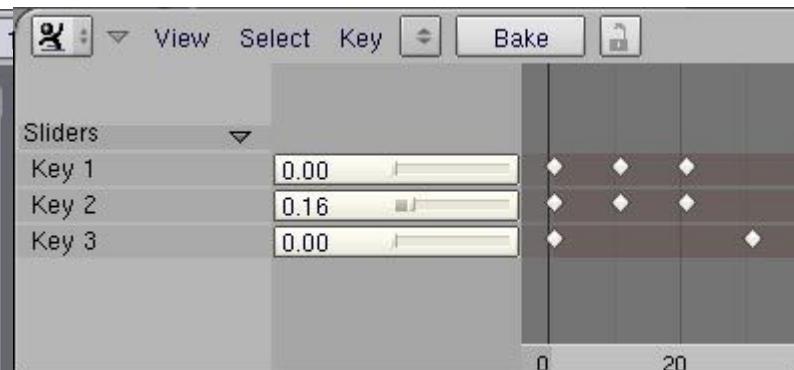


Lattice key sliders in the Action Window [Noob note for blender 2.48a] In the action window, select “ShapeKey editor” in the dropdown “editing modes for this editor” (in the panel header)

To add more Lattice keys, for a variety of deform shapes, repeat the process: I-Key, Tab to Edit Mode, move control points, Tab back to Object Mode. Each time you exit edit mode, a new key will be added to the list in the Action Window.

Animating the Lattice uses the same process as animating Shape Keys for meshes.

If you want the object to begin undeformed, then set each key slider to zero on the first frame of the animation. Move through the frames, setting the sliders as you go to deform the Lattice as desired. You will often find you'll need to set a key before and after each deform key in order to control the rate at which deformations take place in the animation.



Setting keys in an animation
Noob note: I'm doing everything as described and get no keys in action window when exiting Edit mode... Any suggestions?

Noob note; Is it possible to create a keyframe using the object's mesh rather than applying a lattice?

Noobie note: What to press after I-Key? I'm not getting any keys in the action window, help please.

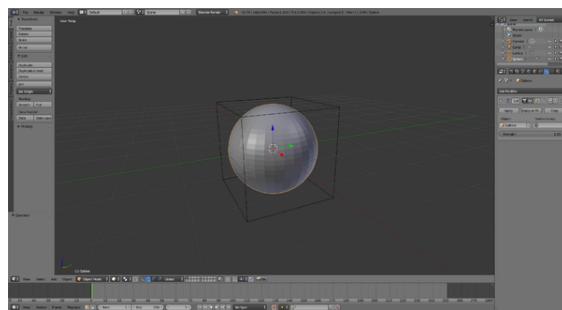
Another Noob: No keys here. I got something changing to IPO Curve Editor and choosing Shape in IPO Type

Noob note; Select Action Editor header->Shape Key Editor. Make sure you also selected Buttons Window->Editing (F9)->Relative Keys

Noob Tip: I am seeing a possible bug in 2.49 in which the key sliders will not appear on the Action Editor unless there is at least one window set to Action Editor when the keys are created. To reveal key sliders for keys that have already been created, save the file, re-open it, set one window to “Action Editor” set the window sub-type to “Shape Key Editor” then switch the Lattice “Relative” setting on and off (putting the cursor over the Action Editor window and hitting HOMEKEY will help locate the sliders). Or, by preference, make sure at least *one* window is set to “Action Editor” before creating the keys.

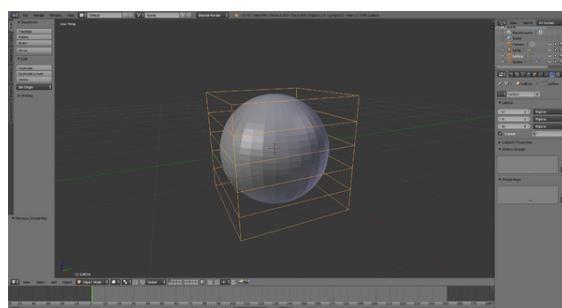
Noob note: I couldn't see the sliders either; then I scrolled up and there they were, off the visible portion of the window. However, I had to then change the window type a few times, so I can't rule out that that did something first.

Creating Shape Keys



The starting point for lattice animation tutorial.

We are going to animate the lattice using **Shape Keys**. Open a new Blender file, get rid of the default cube, and add a new UV Sphere with 32 segments and 32 rings. Then add a new Lattice and upscale it 2 times so it becomes visible (S 2KEY Enter). Scaling the lattice does not have any effect on shape of the sphere *until the lattice is deformed*. When adding new objects the 3D Cursor should be in the same spot (centering your cursor using Shift + C is recommended). Consequently the sphere and the lattice will have the same center. Finally, select the sphere, add a Lattice modifier, and select previously created lattice as object.



Setting the lattice parameters.

For the sake of simplicity we will only deform the sphere

by making it wider or narrower. To achieve this, we will need to give our lattice more vertices that can be tweaked by adjusting the UVW parameters. In the Lattice context

set the W to 6. Also check the “Outside” checkbox to have vertices only on the edges of the lattice cube (again, for the sake of simplicity). *Lattice parameters have to be set **before** adding any shape keys. You will not be able to change them later.*

Before deforming the lattice, add the “Basis” shape key:

1. Select the lattice.
2. Go to the Lattice Object Data context in the Properties window.
3. Under **Shape Keys** press the "+" button and the “Basis” key will appear (note that the UVW parameters are grayed out now).

The “Basis” key represents our starting point (shape). When no other shape keys affect the shape of the lattice (their values are 0), its shape is only governed by the “Basis” key.

Now let's deform the sphere with additional shape keys to be able to animate the deformation later. The basic procedure is as follows:

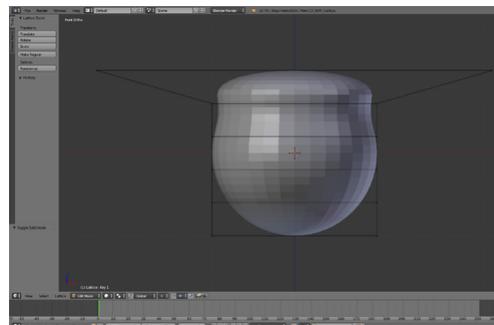
1. Have the lattice selected in **Object Mode**.
2. Add a new shape key by clicking the "+" button once more. “Key 1” or similar should appear in the list. Naturally, you can rename the shape key by double-clicking on its name and entering a new one.
3. Go to **Edit Mode** and deform the lattice to your liking. *You are editing the shape key that is currently selected.*
4. Go back to **Object Mode**.

You have probably noticed that your shape keys appear to have no effect when back in Object Mode. This is because their values are set to 0 by default. To change how your sphere really looks, select a shape key you have previously defined in the Lattice context , and set the “Value” slider to, for example, 1. If you have multiple shape keys defined, you may also “mix” them by setting multiple shape key values to non-zero. There are two more things to note:

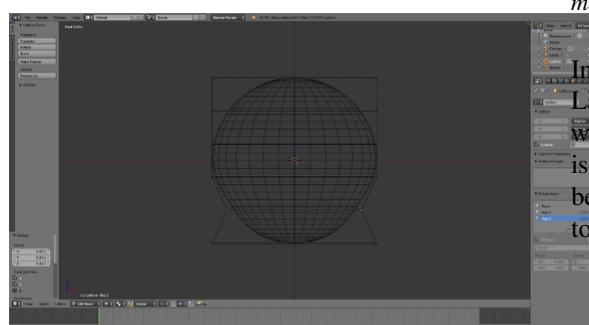
- The “Basis” key does not have a value. It is a starting point after all.
- In Edit Mode you edit shape keys one by one (separately). You could imagine that the value of the currently selected shape keys gets set to 1 and all other values are set to 0 while in Edit Mode.

In this example we will create only two shape keys (beside the “Basis”) and perform two simple deformations on the lattice (which will consequently deform the sphere, since the Lattice modifier is active), which will result in a goblet-like shape. Create a new shape key “Key 1” before deforming the lattice further. This way you will edit the new key instead of the “Basis”. Go to Edit Mode and switch to front orthographic view (keys NUM1 and NUM5). Use Border Select (B) to select the top row of lattice vertices and upscale them in the XY plane (or non-Z directions, use S Shift + Z). Go back to Object Mode, create shape key “Key 2” and, as shown before, downscale the second row of lattice vertices from the bottom. When back in Object Mode, you can set the value of both shape keys to 1 and observe the combined deformation of both keys. This will be our final shape in animation.

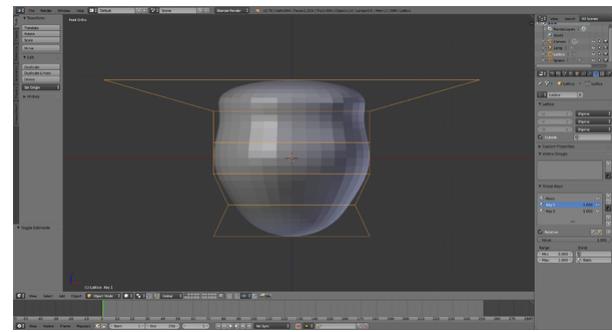
Note that there is no “Limit selection to visible” button  in Edit Mode when editing a lattice. When using Border Select or similar tool, the vertices are selected even if they are not visible, so there is no need to switch to wireframe view to be able to select whole rows in the Front view.



- the first shape key.

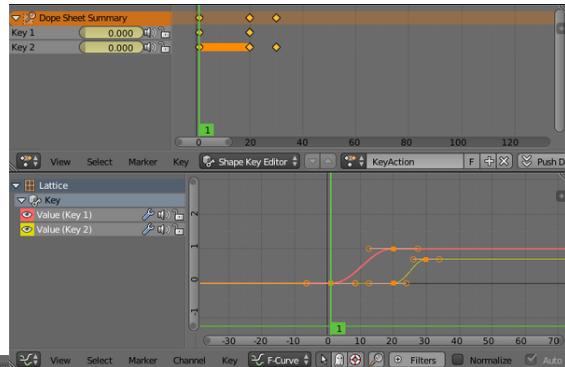


- the second shape key.

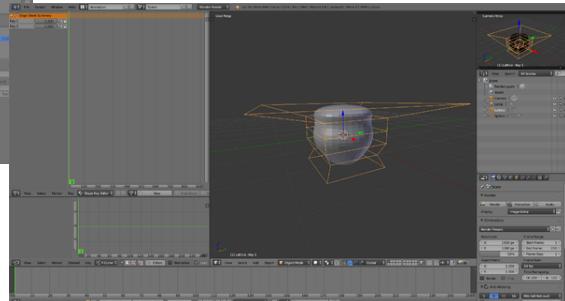


The result with both shape key values set to 1.

Animating Shape Key Values



Shape Key Editor context of Dope Editor (above) and Graph Editor (below). Screenshot does not correspond to the tutorial.



The “Animation” layout. Dope Sheet context is set to “Shape Key Editor” and shape key values are visible in the Dope Sheet Summary.

In previous steps we have created our objects, added a Lattice modifier, and defined the shape keys. This is all we need to proceed to shape key animation. The concept is very similar to animation of any other values. We will be, however, using one additional tool. Follow these steps to set up the interface you will need:

1. **Opening** new Dope Sheet  window or, even better, switch your screen layout to “Animation”. Examples use the latter method.

2. In the Object Mode of 3D View select the lattice object.
3. Switch the context of the Dope Sheet window to “**Shape Key Editor**” using the box in the window’s header. Your shape keys should show up in the **Dope Sheet Summary**.

Shape key values have grey background. This means they are not animated (yet). We will make the sphere deform from its original shape (“Base” key) to the goblet-like shape (influenced by “Key 1” and “Key 2”). Before going forward with examples, we will learn the basic keyframing procedure for shape key values.

1. **Set the frame number.** Either position the green bar cursor in the Dope Sheet to the desired frame using LMB , or use controls in the Timeline window



. For more information about frames see Basic Animation.

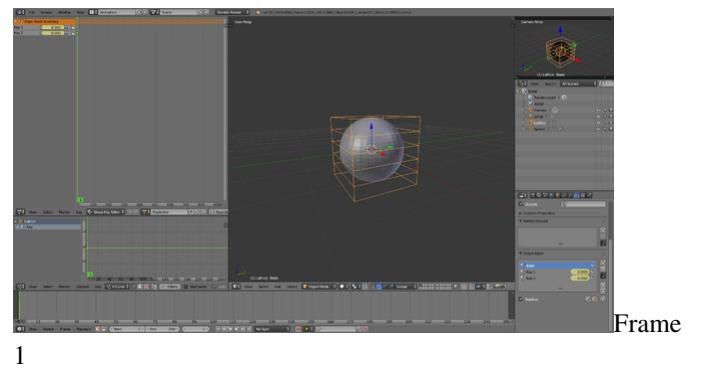
2. Change the shape key values in the **Dope Sheet Summary** either by pressing LMB and dragging, or clicking on the slider and typing the desired value. Notice that Blender inserts a keyframe for changed values as soon as you do (indicated by yellow diamonds in the Dope Sheet). Keyframed values should change their background colour to yellow (means that the value is keyframed *in the currently selected frame*) or green (means that the value is keyframed *in some other frame*).

3. Once the keyframes are added, values of keyframed shape keys will be shown in the **Graph Editor** as well, allowing you to fine-tune them to your liking using F-Curves.

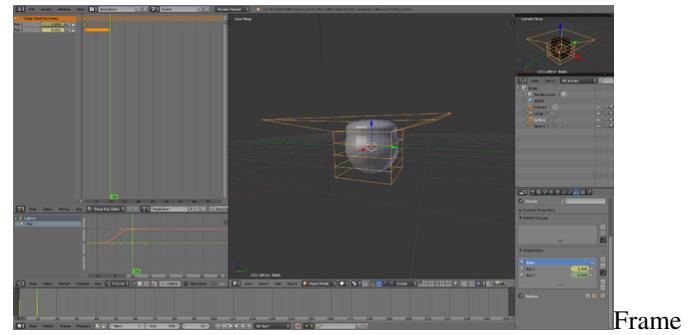
Alternative method is to insert keyframes in the Lattice Object Data context . Select a Shape Key and adjust its value using the slider below. Then RMB the slider and select “**Insert Keyframe**”.

Now let’s animate the sphere using this procedure. Make sure you see the Shape Key Editor context of the Dope Sheet and have your lattice selected in Object Mode of the 3D View. Shape keys should be shown in the Dope Sheet Summary. Set your frame to 1. This is the first frame of our animation (indicated by the “Start” field in the Timeline). In the Dope Sheet Summary set both shape keys values to 0 (note the diamonds -- keyframes -- appearing). This means that in frame 1 our sphere will be shaped only by the “Basis” key. Go to frame 10 and set the value of “Key 1” to 1. The following will cause the top of our sphere to deform during frames from 1 to 10. Do not forget to insert a keyframe for “Key 2” (value 0) at frame 10. To do this just set the value to 0 again or RMB → *Insert Keyframe*. Dope Sheet should show an orange

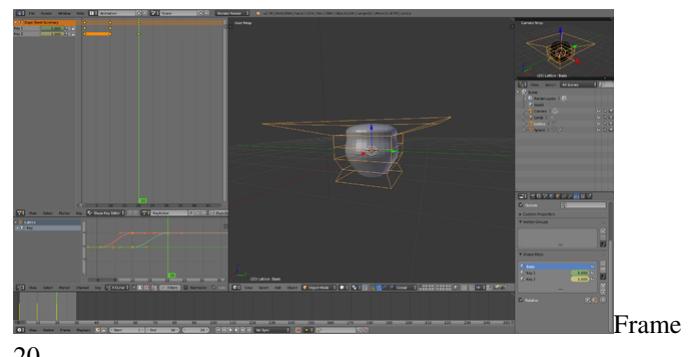
line connecting the two diamonds, showing that values in these keyframes are the same (nothing is happening with this shape key). If you do not add this keyframe, the bottom will start deforming at the beginning of the animation. Meanwhile you may use the mouse wheel or drag the ends (dots) of the sliders in the Dope Sheet and Graph Editor windows to zoom in the view. Go to frame 20 and set the value of “Key 2” to 1. Similarly, this will cause the bottom of the sphere to deform during frames from 10 to 20 *while the top is already deformed*. Finally, set the “End” field in the Timeline to 30, which will stop our animation at frame 30. Feel free to play your new animation using controls in the Timeline. Also feel free to add additional keyframes and tune the transitions to your liking in the Graph Editor. In addition, you might want to try scaling, moving, and rotating the *deformed* lattice (the situation at frame 30 for instance) in Object Mode of the 3D View to see what happens to the sphere.



● 1



● 10



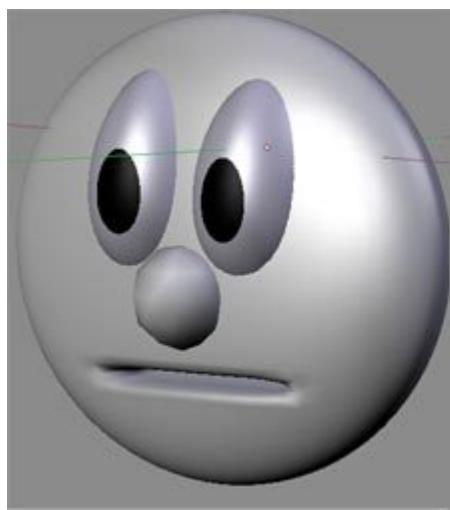
● 20

3.16.7 Let the fun begin!

Now you have your object safely locked away inside your Lattice, you can still animate the object itself, inside the Lattice!

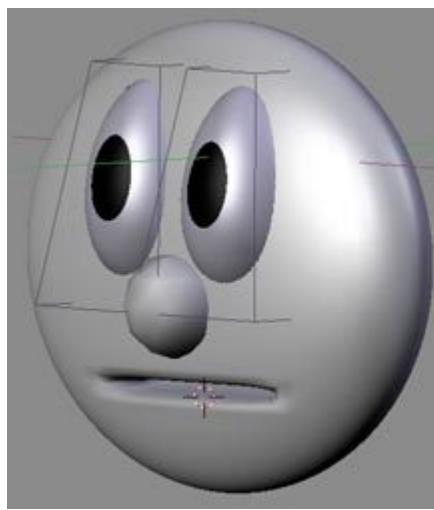
Why would I animate the object too?

Take the case of the classic cartoon eyeball which is taller than it is wide. If you just stretched the sphere object itself, instead of using a Lattice to deform it, then you couldn't properly rotate the eyeballs to look up and down because the whole "egg-shaped" eyeball would rotate and end up lying on its front or back. The eyes would literally pop-out of the head.



Cartoon eyes: The stretched spheres don't rotate properly.

If, however, you use a Lattice to deform the basic sphere to make it into a tall "egg" shape, you can still select the eyeball itself and animate it within the Lattice. Now when you rotate the eyeball, it will look up and down but still maintain its deformed shape within the head.



Cartoon eyes: Spheres deformed with Lattice can still be rotated with good results.

3.17 Bouncing Ball with Lattice

3.17.1 How to make a ball bounce convincingly!

This tutorial assumes basic Blender awareness and some knowledge of using Blender Lattices as well as keyframes basics regarding the Blender Panels. Knowledge of IPO curves will prove useful later as we progress to more advanced techniques.

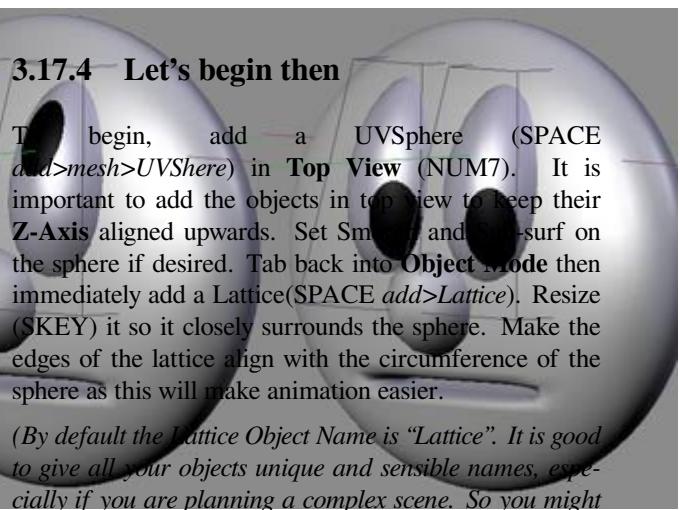
3.17.2 How hard can it be to move a ball?

It would be easy to just put a sphere on the screen and animate it to move up and down but in all honesty, it would not look like a bouncing ball. It would not be convincing in any way.

To be believable, the ball must use some of the most fundamental principles of good animation. In particular, the ball must **squash** and **stretch** and change **speed** as it falls, hits the floor, bounces and rises ready to fall again. With a little effort we can make that boring sphere look alive!

3.17.3 Why use a lattice?

It is possible to make a simple bouncing ball animation without using a Lattice object but with the Lattice we can do more than just bounce the ball. For example, once we have a ball that bounces how we'd like it, we can later add rotation to the ball so it spins through the air and bounces then, as the bounces decrease, the ball can roll to a halt. Doing this without lattice would be a far more complex exercise as a rotating squashed and stretched sphere would look like it was wobbling in space (*like having the flat part of a flat tire spin around the tire instead of staying on the bottom, even while the wheel is turning*).

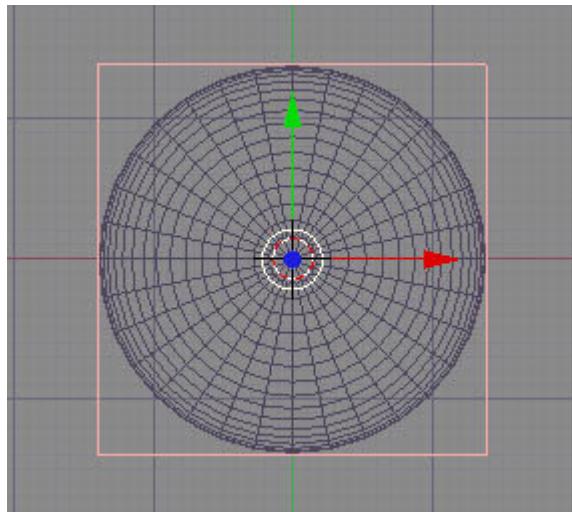


3.17.4 Let's begin then

To begin, add a UVSphere (SPACE *add>UVSphere*) in **Top View** (NUM7). It is important to add the objects in top view to keep their **Z-Axis** aligned upwards. Set Smooth and Subsurf on the sphere if desired. Tab back into **Object Mode** then immediately add a Lattice(SPACE *add>Lattice*). Resize (SKEY) it so it closely surrounds the sphere. Make the edges of the lattice align with the circumference of the sphere as this will make animation easier.

(*By default the Lattice Object Name is "Lattice". It is good to give all your objects unique and sensible names, especially if you are planning a complex scene. So you might call the Sphere "Red_Ball" and the Lattice "Red_Ball_Latt" or something similar for easy reference and recognition*

later.) (Noob Tip: I have formed the habit of naming everything in ALL CAPS. As the scene gets more complicated I can easily keep track of what I have named and what needs naming.)



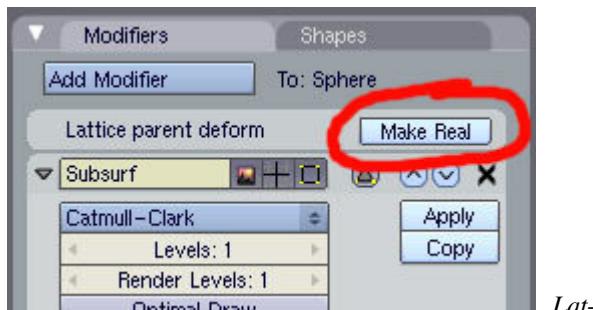
Top view of UV-Sphere inside selected Lattice

Make the lattice deform the sphere

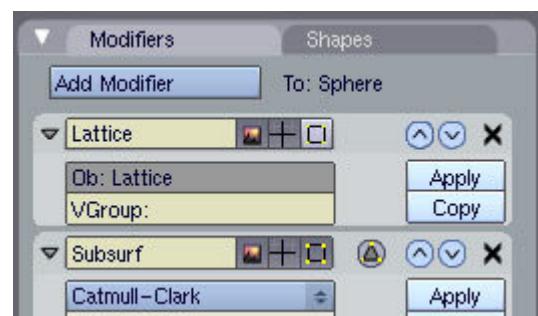
Prior to Blender 2.4, objects were parented to Lattices with a Lattice Deform option. In version 2.4+ you can select the object (the sphere) and apply a Lattice Modifier to it (Ob: Lattice). However, if you use this new method then the ball will not move where the Lattice moves and will not squash when the Lattice squashes and we want it to do both of these things.

Luckily you can still use the good old **Parent option** then use the Modifier palette to make the Lattice Deform "Real". So select the sphere and THEN SHIFT SELECT the Lattice then **CTRL-P** to make parent and choose the "**Lattice Deform**" option. If you now select just the sphere, press **F9** and look in the Modifiers palette, you'll see a listing for "**Lattice parent deform**" with a button for "**Make Real**". You can either ignore this button or press it. It really won't matter for now. If we wanted to do some weird things to our ball, like kicking it or deforming it in other complicated ways, then we would have to use the "real" Lattice modifier option.

So, this may not be the purist approach to using a Lattice but it is a convenient way to achieve the results we're after. Basically, it works and is reasonably intuitive.



tice Modifier screen using parenting option.



If you press "Make Real" you enable the full Lattice Modifier (None of this seems to be necessary in later versions. The full Lattice screen comes up as soon as you Parent it.)

You can easily check if the Lattice is working the way we want it to by selecting the Lattice and resizing, moving or rotating it. **The ball should do everything the Lattice does.** I don't wish this to be a full lesson in Lattices so if you don't understand any of the above, or you're certain your Lattice is not working as it should, then please learn a bit about lattices before proceeding.

Time to animate!

For our basic animation, we'll use just 23 frames starting with the ball high then falling, squashing, bouncing and ending up back where it started. When this is played back in the 3D window, it will loop and we'll see the ball bounce forever in one place.

This part is easier done in "SCR:1 - Animation" Viewport configuration. It is also useful to have multiple orthographic views open in the viewport by right clicking on the resize viewport arrow and choosing "split area" then making a top view, side, etc...

Go to **Front View** (Num1) Press F10 and set your start (Sta) frame to Frame 1 and your end frame to Frame 23. Then make sure you're on **Frame 1** (Shift+Left Arrow) to start animating. Note that we are **animating the Lattice**, not the ball. Everything we do to the lattice will directly affect the ball too.

Note: all the modifications to the lattice must be done in 'Object' mode. If not, they will affect all the frames.

Select the Lattice (**NOT** the ball) then do the following on the corresponding frames:

- **FRAME 1:** Press IKEY and set a **key** for LocRotScale - the ball's start position (high).
- **FRAME 11:** Move the Lattice to ground level and key LocRotScale again (*keep the fall distance small for now*).
- **FRAME 13:** Leave Lattice in same position as frame 11 and **key** LocRotScale again.

- **FRAME 23:** Move Lattice back to start position (use Num (N) Panel if you need to) and **key LocRotScale again.**

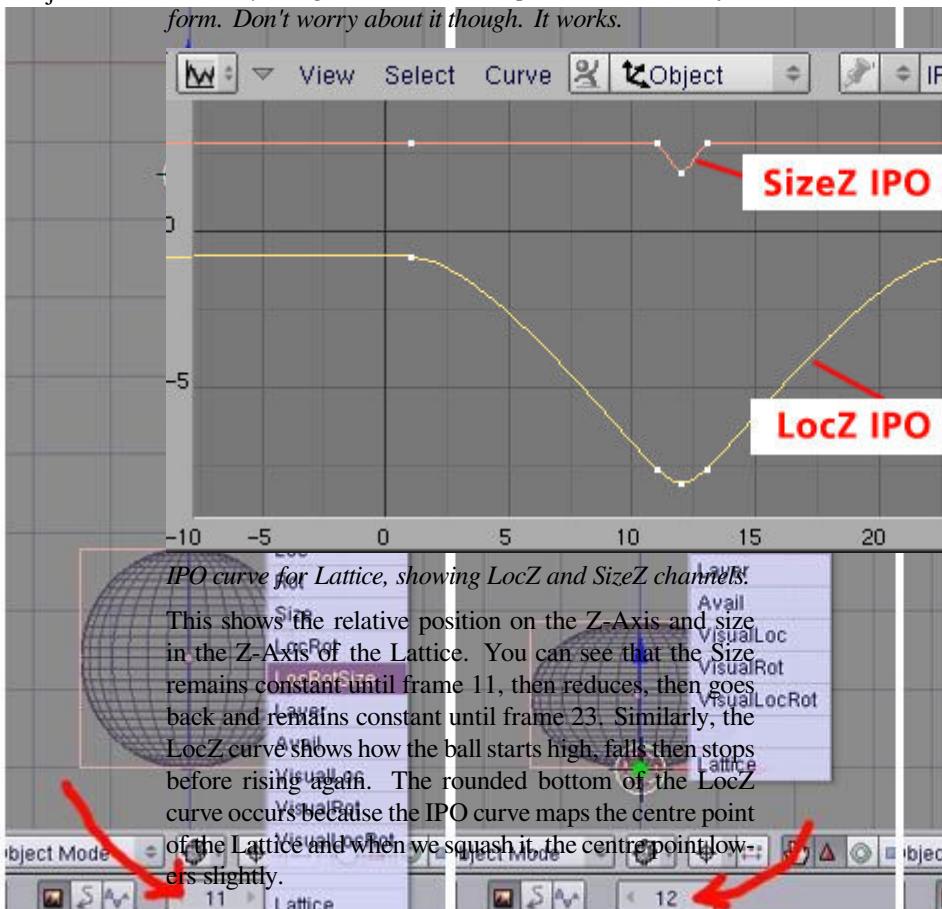
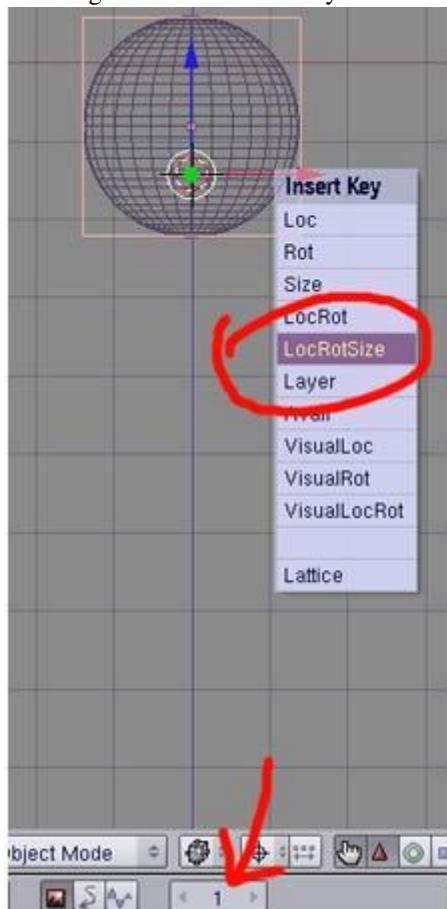
(Noob note- This may seem obvious, but move the lattice BEFORE you insert the keyframe.)

These four frames give a basic up and down animation with a pause at the bottom. We can add squash on Frame 12 to finish off.

- **FRAME 12:** Place the cursor at the base of the Lattice then set the Rotation/Scaling Pivot to “3D

 Cursor” Scale the Lattice down along the Z-Axis **SKEY-ZKEY** and see the ball squash. **IKey LocRotScale again.**

When you have set all five keys, you can cycle through the frames and you should see the following result on the Key frames we just set.



Screenshots of the 3D window, front view, on each of our five key frames.

If you press **Alt-A** now in the **3D window**, you should see the ball bounce. Even though the squash happens on just one frame, it still reads properly in our minds and is a vast improvement over an animation with no squash at all. (*Press RMB or ESC to stop the animation playback*).

If you have a camera and lighting suitably set up you could render the animation out and use Blender’s “PLAY” but-

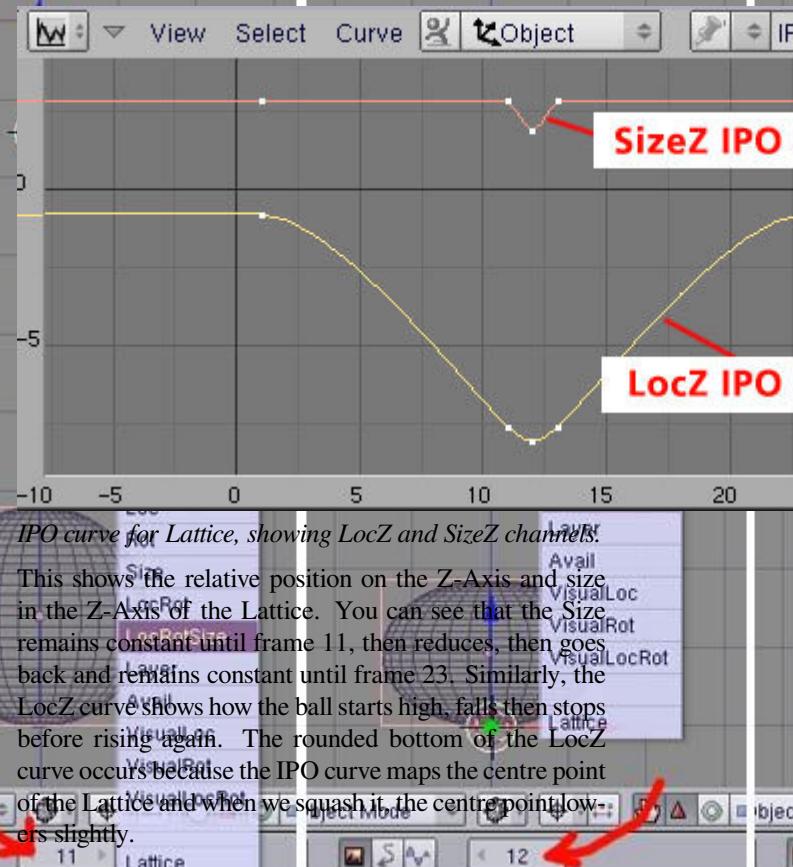
ton to loop the rendered animation. See the bottom of the page for an OpenGL render made from the steps above. Add material to the ball and a ground plane for a more interesting result.

Looking at the IPO Curves

If you want the ball to bounce three times then you could repeat the above instructions three times over - if you have the patience - but Blender has a better, easier way.

Select the Lattice then open up an IPO (InterPOlation) window and zoom into the group of keys. If you select LocZ and SizeZ from the channel list on the right, you should see something like this (I've added labels to the screenshot):

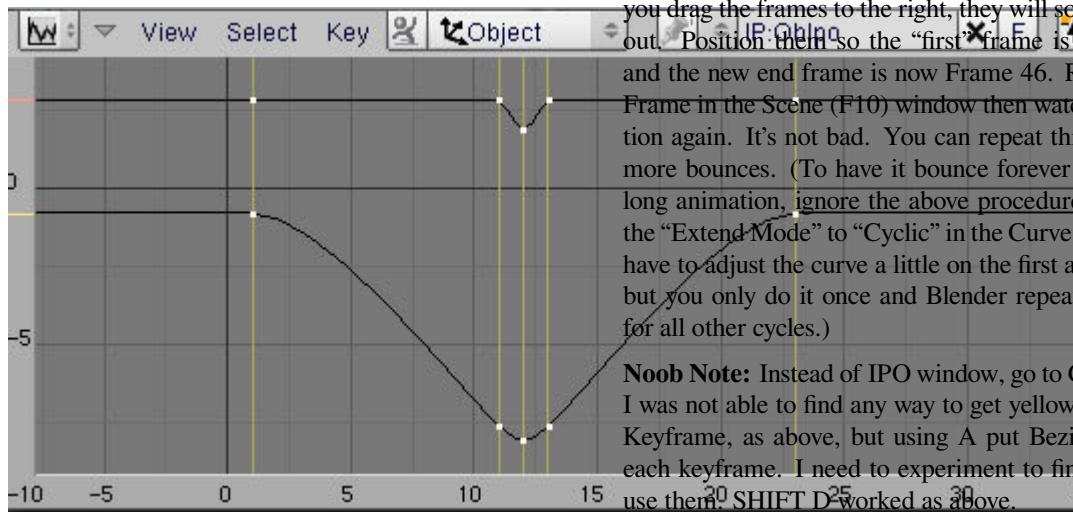
Note the the IPO Type here is “Object”. Purists may be disturbed or perplexed by this but if you remember, we're not really using our Lattice in the purist, Lattice Modifier form. Don't worry about it though. It works.



Of special interest is that by default, Blender uses Bézier curves for IPOs and this gives us rounded tops on the curve. This means the ball starts moving slowly downward then moves faster until it reaches the bottom. Also, as it nears the top, it slows down again. This is good because this is exactly what happens in reality when a ball rises and falls (*It could be better - but it's a good start*).

While in the IPO window, select both curves then press **KKEY** and you'll see each key frame indicated by a vertical line. (Note: You can also press **View->Show Keys** if

you can't remember this shortcut - this has helped me in the past.)



IPO window set to show key frames



Bouncing ball animated GIF

Press **AKEY** to select all key frames then duplicate (Shift-D) them. The curves will seem to deform but when you drag the frames to the right, they will sort themselves out. Position them so the “first” frame is at Frame 24 and the new end frame is now Frame 46. Reset the End Frame in the Scene (F10) window then watch the animation again. It’s not bad. You can repeat this process for more bounces. (To have it bounce forever throughout a long animation, ignore the above procedure and just set the “Extend Mode” to “Cyclic” in the Curve menu. You’ll have to adjust the curve a little on the first and last frame but you only do it once and Blender repeats the change for all other cycles.)

Noob Note: Instead of IPO window, go to Graph Editor. I was not able to find any way to get yellow lines at each Keyframe, as above, but using A put Bezier handles at each keyframe. I need to experiment to find out how to use them.²⁰ SHIFT D²⁵ worked as above.

It's not bad for a start but there's a lot more to do to this ball before applying for that animation job.

Note: In newer versions of Blender, you can go to Graph Editor, Select all graphs with **AKEY**, and select Channel-Extrapolation Mode-Make Cyclic. This will extrapolate the existing graphs to cover the total frames.

3.17.5 Saving The Animation

How do you tell Blender where to save the animated frames? The answer is in the “Output” mini-window that you see in the Render settings (F10). Near the top of this are two editable text fields, the first of which is initially set on my Linux system to “/tmp/” and the second to “//backbuf”. The first one is the output pathname or directory prefix used to save the generated frames as JPEG files when you press the “ANIM” button. If it has no “#” characters in it, then it is used as a directory prefix; thus, the default will be to save the animation frames as /tmp/0001.jpg, /tmp/0002.jpg etc. Alternative, if you put in a filename spec with “#” characters in it, the hashes will be replaced with the frame number. Thus, if I set the field to “/home/lde/Documents/BlenderTuts/ball####.jpeg”, then the frames will be saved to that /home/lde/Documents/BlenderTuts directory and be named ball0001.jpeg, ball0002.jpeg etc.

3.17.6 So what next?

In real life and in a typical cartoon, balls don't just bounce forever in one spot. They move along and slowly lose their bounce before rolling to a halt. And they have more character than our bouncing ball.

Things to consider next:

- Perfecting the squash - maintaining volume

- Adding stretch
- Look at acceleration and deceleration
- Rotate the ball as it bounces
- Reduce the bounce height over time
- Move the ball around as it bounces

See this **ANIMATION LINK** (580kb, AVI-DivX) for an idea of how you can achieve interesting results with just a little more effort. but its k? k

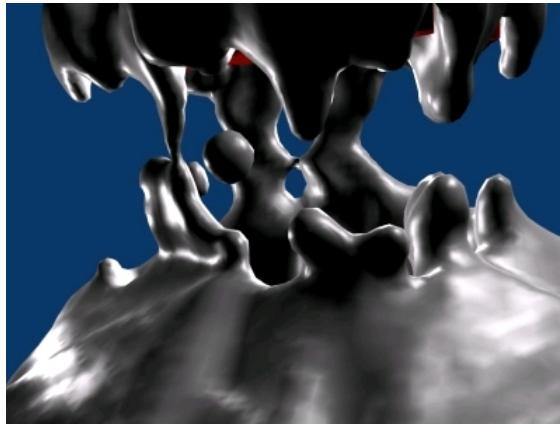
3.17.7 Links

Principles of bouncing ball animation: <http://www.idleworm.com/how/anm/01b/bball.shtml>

Blender 2.49B bouncing ball animation: http://www.youtube.com/watch?v=BEY_6dFdUJY

Blender 2.49B vector blur animation, to add realism to your animations: <http://www.youtube.com/watch?v=qY4WcNgEXv8>

3.18 Creating Basic Water Animation



A low resolution, high viscosity fluid test

3.18.1 Water and Other Fluids

Water is without a doubt one of the most important compounds in our lives; It covers about 75% of the earth and is therefore incredibly important in quite a few Blender animations. Wouldn't it be great if we could get an accurate physical representation of this liquid in Blender? We can, using a tool called Fluid Simulation. This tool looks unnervingly complex at first glance, but this tutorial should clear it up for you. At least to a basic level.

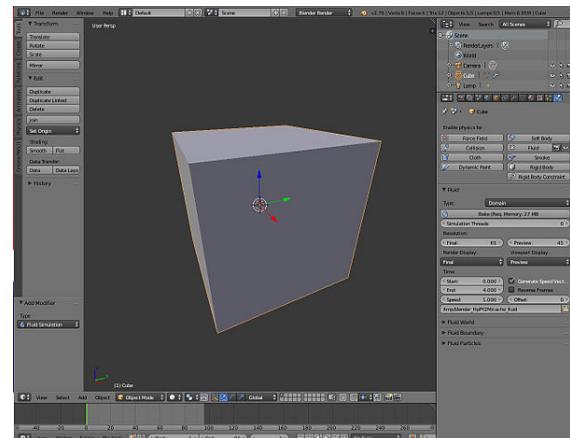
3.18.2 The Domain

One can imagine how much time it would take Blender3D to think about everything in the infinite space of a 3D world in terms of fluid objects and deflection, so we obviously need to cut down on that size. Create a fairly large cube. This will eventually be set as the volume in which all of the fluid simulation occurs. Don't make it too large, but not too small either, let's say 10 times bigger. With

this cube selected, go to the Physics context of the Properties window. Find the button that says "Fluid", and click the "Fluid" button to enable the function. Set type to "Domain". All the fluid physics will be calculated inside this cube. Also in that tab, you'll see 3 sub-sections, "Fluid World", "Fluid Boundary" and "Fluid Particles". Each one opens a different set of settings. These will be explained as they become important.

Under "Time" you can adjust how much seconds the fluid animation will take. Due to this you should change the "End" frame to "96", under the "Timeline", because the standard setting is "4". You can also change it how you like.

To make the other objects go into the "Wireframe" viewport.



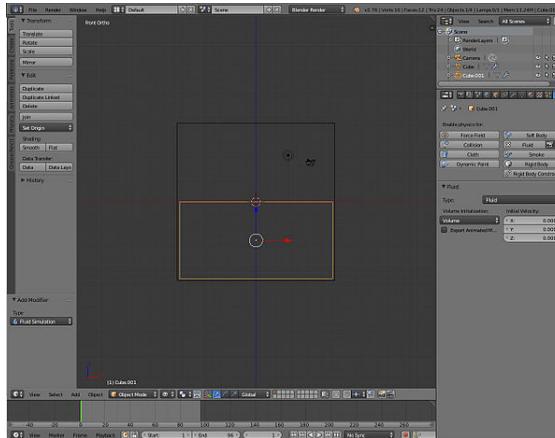
Domain

3.18.3 The Fluid

As a basic start up for your experiments with fluid simulation, I am going to take you through a small demo in which we drop an object into a pool of water, creating a splash. To do this, we need a fluid object and an obstacle object. For the fluid, create another cube, scaled down to cover the bottom of your domain. Make a rectangle, that hovers just inside the domain and is half as high as the domain. Enable this object in fluid simulation also and set type to "fluid". When we start to bake the fluids, this will be set up as your liquid.

Let us go over what you have now. You have your starting cube, with fluid physics and type "domain", and will

be referred to as “Domain”. Inside this is the rectangle that will cover the bottom of your Domain , which is the liquid. The liquid will be a different object with the type “Fluid”.

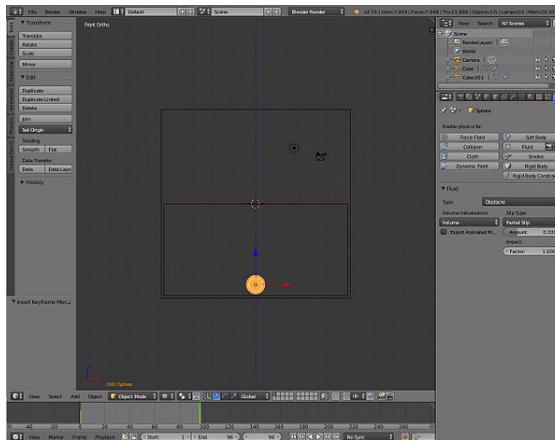


Fluid

3.18.4 The Obstacle

Create a third object which we will call “Obstacle”. This will be the object dropped into the water, and will be described further in the explanation. Give it an IPO/Animation to drop into the water, and enable it as --you guessed it-- an obstacle (click the “fluid” button, in physics, and set type to “obstacle”).

Set up the animation in the front/side views, not the top view. The z-axis is where gravity works.

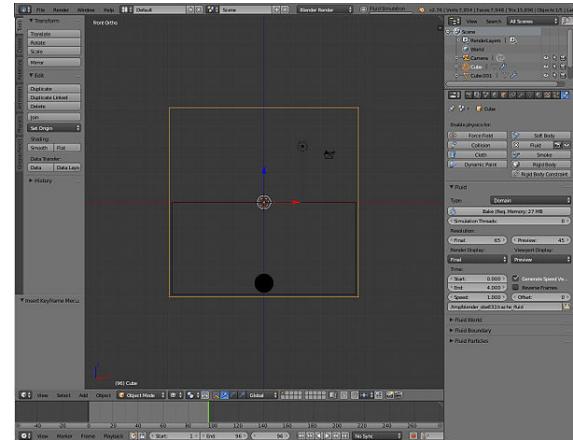


Obstacle

3.18.5 Baking Fluids

Now we get to see what it all looks like. Select the domain object, and find the big button that says “BAKE” under “Physics” - “Fluid”. If you want to use a bake again, don't choose a temporary location, otherwise you'll have

to bake it all over again. Press it. Be patient, blender has to bake this simulation, something like it would render the final product. The loading bar of this baking will stay in the bar at the top of the screen with the blender logo. If at any time you feel that you want to abort this process, press “Esc”. Once the bake is finished, click under the “Timeline” on the play button. You should be able to view the full simulation.



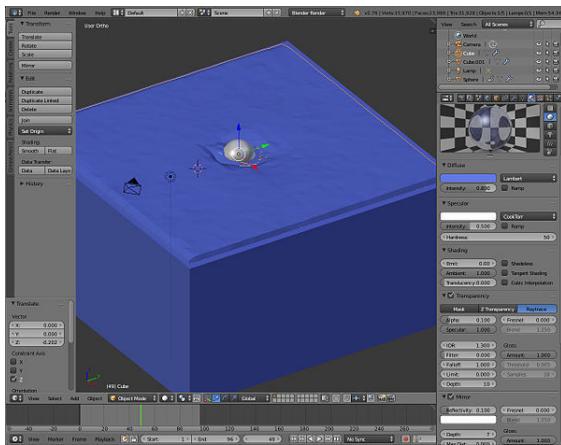
Bake

3.18.6 Finishing touch

Blender will make your water its default opaque grey, unless you set its color. A good way to make realistic, clear water (as well as glass for that matter) is to edit the color, then apply a simple mirror effect plus transparency effect. After your water has rendered, select the domain in object mode. This selects the liquid in the frame you have just added. in the “Materials” tab (if when you click on this, the colors settings are not up, just press add new under the only panel there). Go to the “Mirror” section, play around with the settings until you get what you fancy. Suitable settings include Reflectivity to 0.1 for water, 0.15 for glass, set depth to 7; Under the Transparency section, click on Raytrace and set IOR to 1.3 for water, 1.5 for glass; Then set the Alpha value to 0.1 while adding an appropriate color under the “Diffuse” section, like: “#6279E7”. I should do the same for the “Fluid”

3.18.7 Other Fluid Objects

There are other incredibly useful types of fluid objects like “inflow” and “outflow”. The both of these do exactly what they sound like. Inflow objects pour more fluid into the scene, and outflow objects drain it away.



Simulation

3.19 Flying Through a Canyon

3.19.1 Creating the Canyon

Forming from Textures

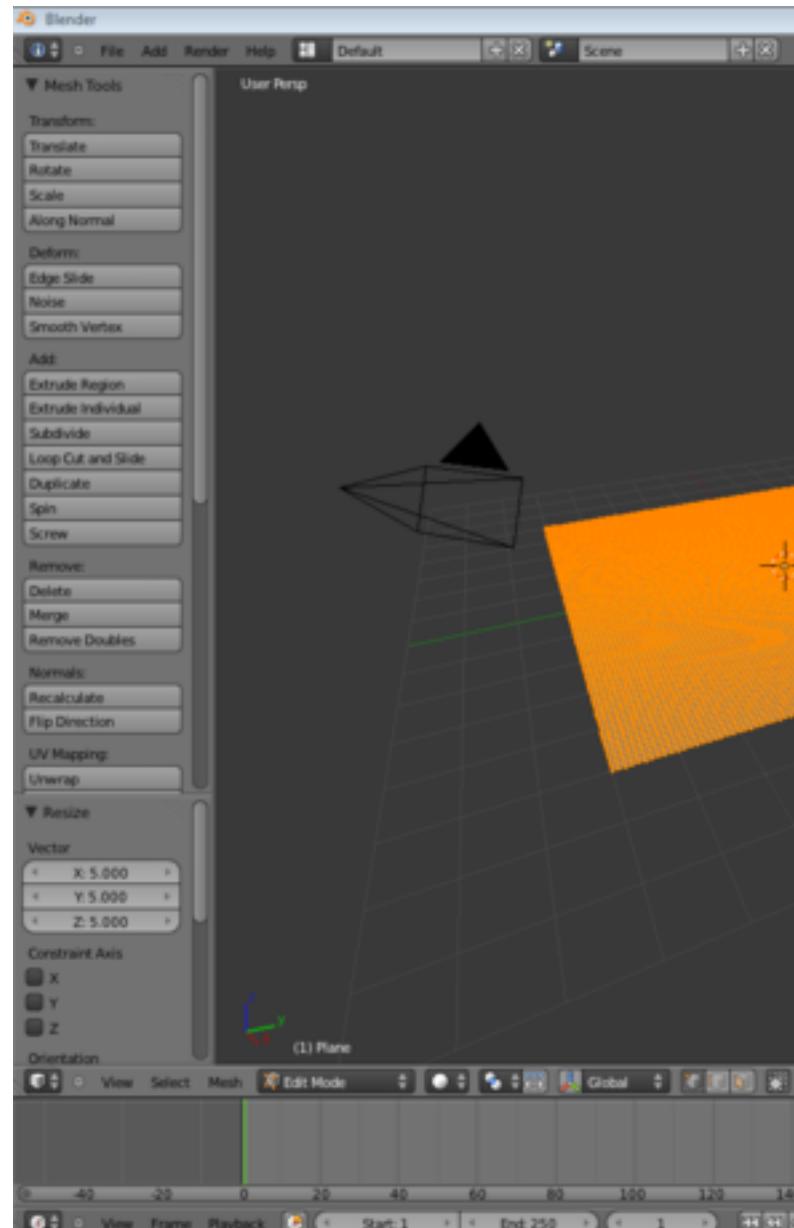
1. Open up a new Blender file and erase the default cube with X

2. Add a grid and change the subdivisions to 100 for both X and Y.

1. Scale the canyon up to 5 times its current size

(Do this by either hitting S and typing in 5, or hitting S while holding CTRL to snap and scale your mesh 5 times.)

At this point your mesh should look like something like the picture below.



Texturing, Shading, and Deforming Your Mesh

- Go to the Texture tab next to the Materials, and click New.
 - This texture will be used to create our canyon, so give it an appropriate name. In this tutorial, “Canyon” was used.
 - Select the Texture Type to be Wood
 - If there is a Checkbox next to “Canyon” in the Texture tab, uncheck it.

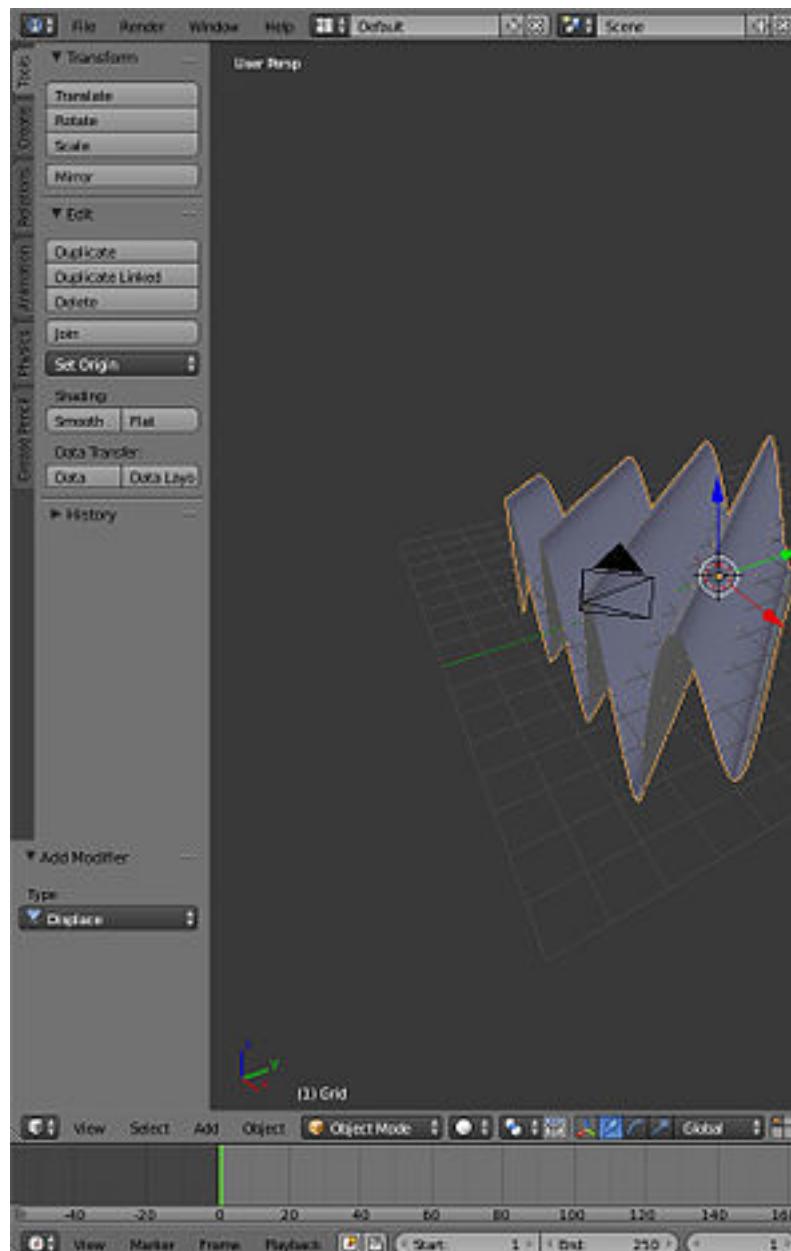
We will now displace the vertices based on the texture.

- Click on the Modifiers tab on the Properties panel (spinner icon).

- Click the *Add Modifier* pop-up menu and select “Displace”.

- Select your “Canyon” texture for the displacement.

- Go into Object Mode. Notice that the vertices have been displaced! (To see the results better, make sure you're in Solid Mode) However, the material is very jagged. This is because Blender is preserving the original faces on the underlying material, and the original faces don't run in the same direction as the grooves the wood material has created.



Improving the Look

Now to make the canyon more interesting:

- Go back to the textures tab, under the Wood tab de-select Bands and select RingNoise. Keep the default options for the moment.
- You will now see a “rippling” texture. But it isn't a canyon yet.

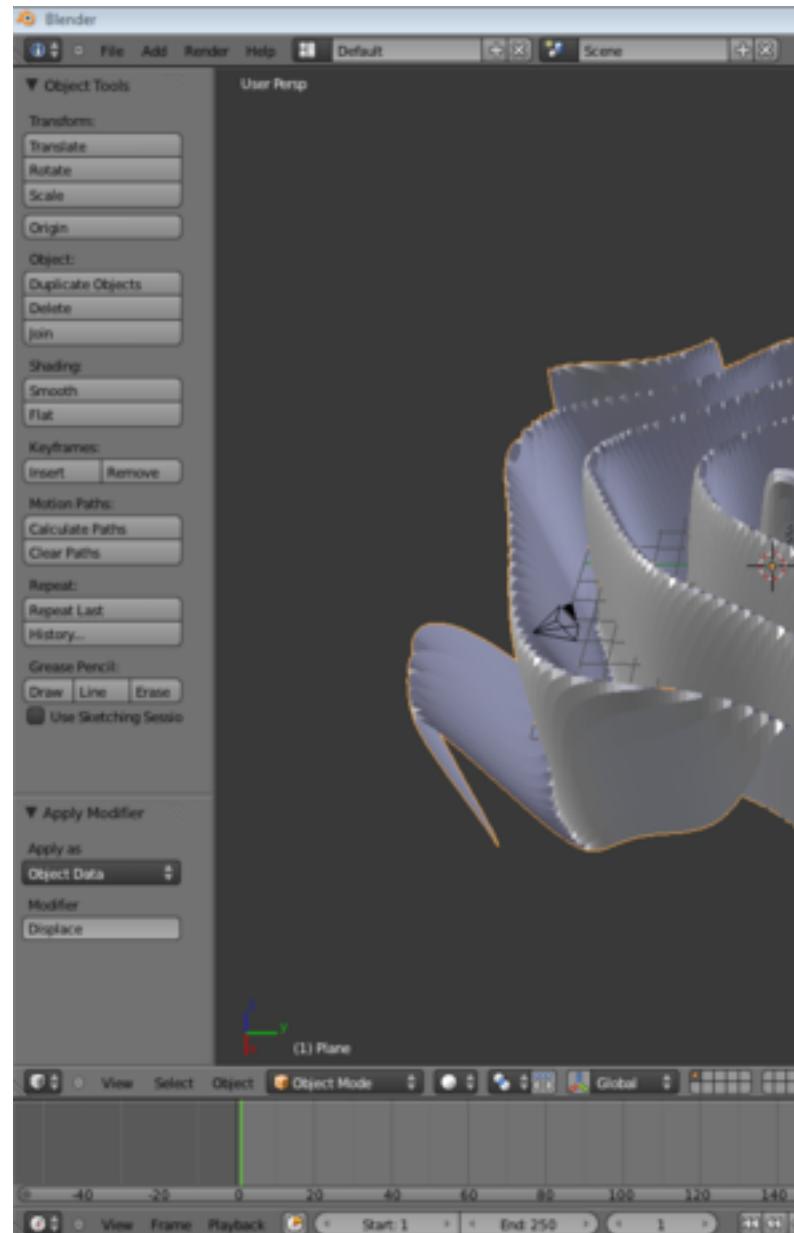
Noob Note: This will also work with an image texture, i.e. using an image as a texture. Tested using NASA's Blue marble topographic pictures. Go to the end of this tutorial for additional projects.

Your Mesh should now look something like the picture below.

Note: If you go into Edit mode (Tab), and select all vertices, then scale to 0.2 of the original size. after this should leave Edit Mode into Object Mode, and scale the Object (as a whole) up to 10.0 of the size. What will occur is that you'll pack the vertices closer to each other, but not the texture. Then, you would have taken the object as

a whole (including the Texture) and up-scaled it. This is the difference between Scale in Edit Mode and Scale in Object Mode. Don't do it!

- You should now have a semi-boring canyon, but it does have circuits which the camera can float around in.
- To make the Canyon more interesting, play around with the Texture values for Wood. I chose the values Noise Size: 1.0 and Turbulence: 15.00 as they produced a somewhat interesting outer ring.
- You may also want to play around the Deform modifier settings, especially *Strength* which changes the strength of the deformations.

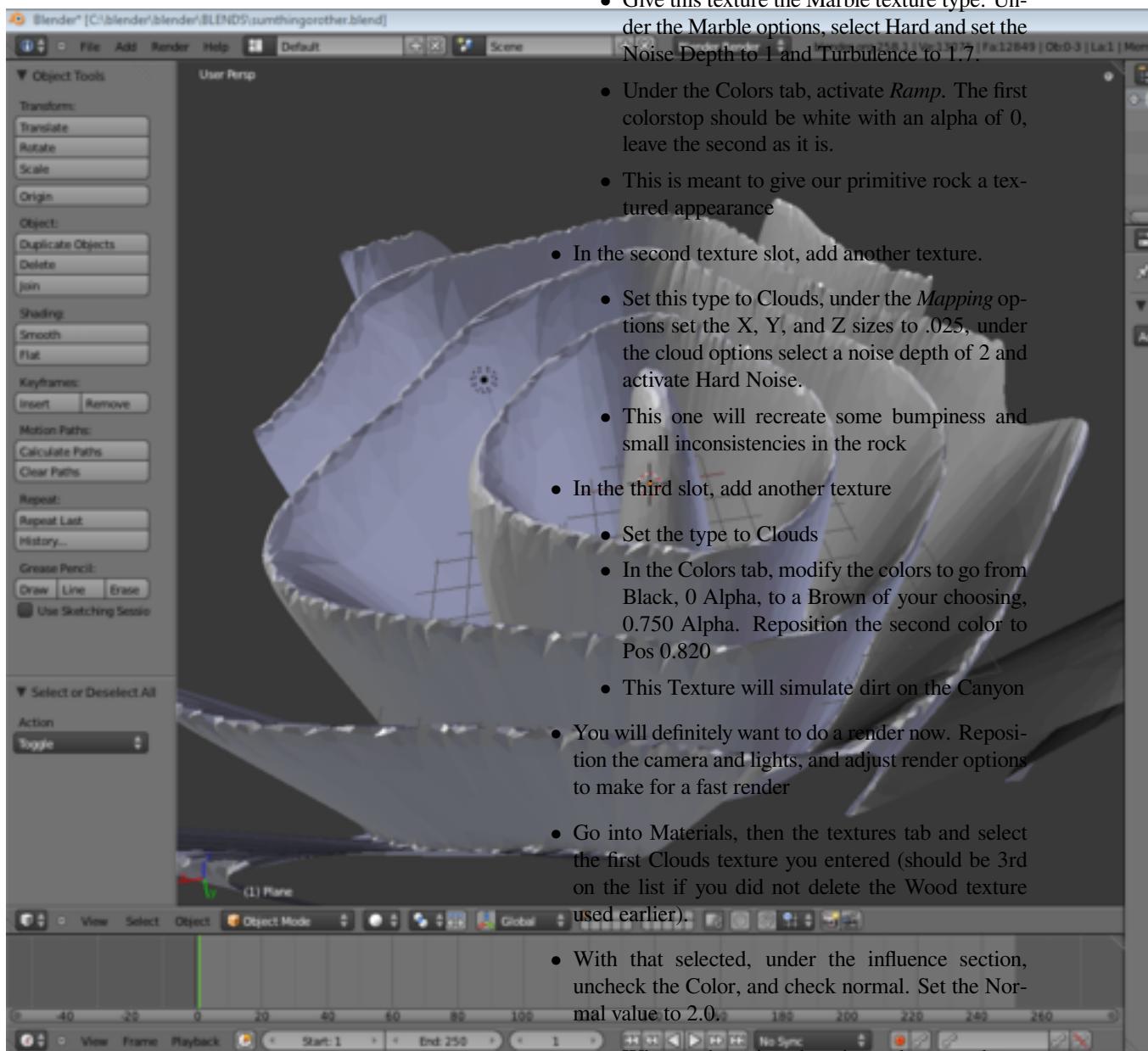


Eliminating the Unneeded

- When you are happy with your mesh, press the *Apply* button under the Deform modifier.
- The displacements are now permanently baked into the Mesh, and the result should be something like the picture below.

- We will now reduce the number of vertices. Go into Modifiers, and add Decimate. The mesh will appear to change slightly, but it hasn't, except that 4-vertices faces have been triangulated
- Add another Modifer, Sub Surf, and make sure that the Levels is just 1
- Under the Decimate, slowly reduce the ratio and watch the Mesh. Very tiny mesh details will be changed but the Decimate modifier will keep the shape while reducing vertices.
- I got to 0.07 on my mesh before I decided the detail was getting too low. Remember you will be flying through the canyon, not over it, so the detail on the walls of the canyon is more important.

- Hit apply to bake the reduced vertex count into the Mesh. Smaller vertex counts will make rendering faster.
- Increase the number of levels on your Sub Surf modifier and take a good look, you have just created a canyon! You may reduce the levels or remove the Sub Surf entirely afterwards.



- Create a new material for the cliff
- In the Specular panel under the Materials tab, set the Intensity value to 0.132 (our cliff isn't particularly shiny) and Hardness to 20.
- Under the Textures tab, add a new texture in the first slot.
 - Give this texture the Marble texture type. Under the Marble options, select Hard and set the Noise Depth to 1 and Turbulence to 1.7.
 - Under the Colors tab, activate Ramp. The first colorstop should be white with an alpha of 0, leave the second as it is.
 - This is meant to give our primitive rock a textured appearance
- In the second texture slot, add another texture.
 - Set this type to Clouds, under the Mapping options set the X, Y, and Z sizes to .025, under the cloud options select a noise depth of 2 and activate Hard Noise.
 - This one will recreate some bumpiness and small inconsistencies in the rock
- In the third slot, add another texture
 - Set the type to Clouds
 - In the Colors tab, modify the colors to go from Black, 0 Alpha, to a Brown of your choosing, 0.750 Alpha. Reposition the second color to Pos 0.820
 - This Texture will simulate dirt on the Canyon
- You will definitely want to do a render now. Reposition the camera and lights, and adjust render options to make for a fast render
- Go into Materials, then the textures tab and select the first Clouds texture you entered (should be 3rd on the list if you did not delete the Wood texture used earlier).
- With that selected, under the influence section, uncheck the Color, and check normal. Set the Normal value to 2.0.
- What we have just done is set the second texture not to affect color, but to affect the angle of the normal in a render. This can be used to reproduce jagged edges on flat surfaces, or waves in water

3.19.2 Coloring the Canyon

A Rocky Material

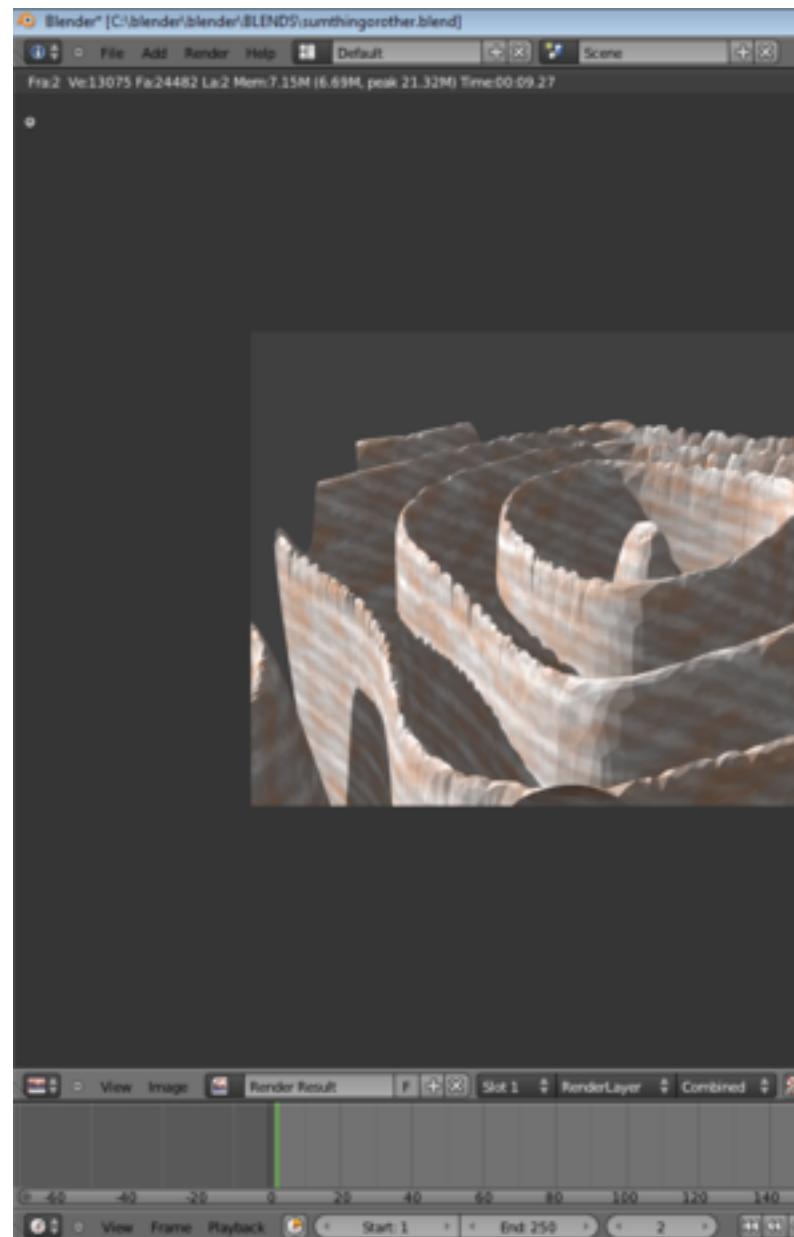
This is one method to create a rocky wall. The result isn't particularly realistic but it is easy to make. Creating materials is a difficult practice, and the first step is to know what you're trying to recreate, perhaps by using actual photographs.

- Now, select the Marble texture again. Under the Map Input, adjust the sizeX, sizeY, and sizeZ values to 5.0.
- Change the Z size under mapping to 5.0 from the second cloud texture.

- You will want to render here, adjust the values as desired. For more layers vertically, increase the sizeZ value
- You should now have an unrealistic, but sort of rocky Canyon wall.

Lighting the cliff

- Remove all existing lights
- Creating a new light with the type of “Hemi”. Center it over our cliffs, and use Scale (s) to scale it down.
- Create another light, with the type of “Sun”. Position it where you want (position doesn't matter with Sun), and use Rotate (r) to position its Ray at the Angle you desire.
 - In this example, the Sun was more of an evening Sun. Its light is at a 45 degree angle on one side. This creates some Canyon walls light, some dark.
- Position your camera and adjust the Energy values of your Hemi and your Sun. The Hemi will light up everything, its an easy way to give light to canyon walls that would otherwise be completely black. As a starting point, set your Hemi value at 0.4 and your Sun value at 1.5 and observe how that looks from a few angles. Make adjustments if you desire.
- Finally change for all the three textures the Mapping - Coordinates to “Generated” in stead of “UV”



3.19.3 Guide the camera through the canyon

The last step is to guide the camera through the canyon.

1. Click on the camera in the Outliner.
2. Click on Num0 and zoom a bit, then lock camera to view.
3. Go to keyframe 1 and set a locrotscale keyframe.
4. Click on the red dot under the Timeline, which will keyframe every movement.
5. For simple keyframing set the Framerate to 8 FPS and set the End frame on “1000” then hit the play button under the Timeline.

6. Now fly with shift+F through the canyon.
7. After this set the frame rate back at 24 FPS and watch and see your beautiful simulation, which will possibly take too long to render.

3.19.4 Additional projects

1. Using NASA's topographic image map at http://earthobservatory.nasa.gov/Features/BlueMarble/BlueMarble_2002.php

You have the option of selecting a small portion of your selected topographic map or you can map the entire 360x180 degree panoramic map onto a sphere. Refer to UV Mapping tutorial.

If you just select a small portion of the map of the world(cropping), note the resolution of your cropped image. Create a basic plane and do not scale it yet. Sub divide as many as required to match the resolution of your image. This is in order to get the most out of the topographic image. If you don't bother, this is not essential.

Then apply the texture. Choose texture of type "image" and do not disable this texture. NASA's topographic pictures also come in a colourful form but shaded to indicate heights. The reason why you leave the scaling to the last is that the texture is set to map onto the basic plane size of 2x2 units. If you increase this, the textures will be repeated. Adjust the strength level to a suitable level. The default 1 is too large.

3.20 Using the Sequencer to Compile Frames into an Animation

3.20.1 Preamble

When you render an animation, often it's a good idea to render the individual frames and then compile them into the animation. This has several benefits over rendering directly to a movie file. Some of them are these:

- If you discover a mistake in the render, you can modify sections of the animation and rerender those frames without having to rerender the whole animation.
- If the render crashes because the power went out or, (psst), blender crashed, you can pick up the render at the last rendered frame without having to rerender the whole animation

3.20.2 Tutorial

- Open your animation in blender

Begin by opening blender to an animation that is ready to be rendered.

- If you don't have one and don't want to wait to long, just download the demo file from [blender.org For You](http://blender.org/For You)
- The output format should stay PNG. Though you can use any image format you want, I recommend PNG because it is lossless format.
- Set the file output location.

Set the location on your file system where you want all the frames to be saved to.

- Set Endframe to match the animation

If you animated 500 frames, set the end frame to that same number. the default is 250.

- Render the animation!

Simply press the "Render" button under the render tab.

- For "For You", you only need to click Num0 and go to "Render" in the menu bar. Then click on "OpenGL Render Animation". For exercise, 100 frames must be enough.

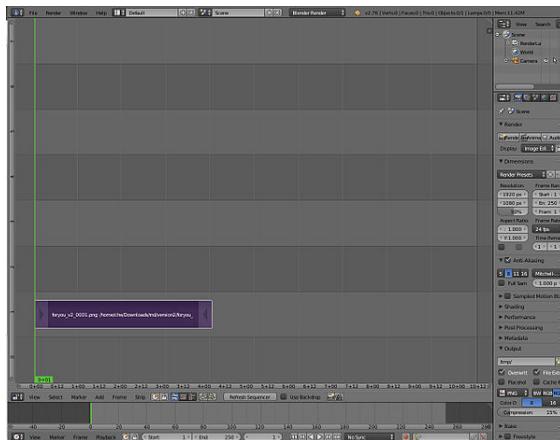
3.20.3 Use the Sequencer to compile the images into a movie file

- Open a new blender and delete everything except the camera.
- Change the 3d window into the "Video Sequencer Editor" window.
- Add the images to the sequencer.

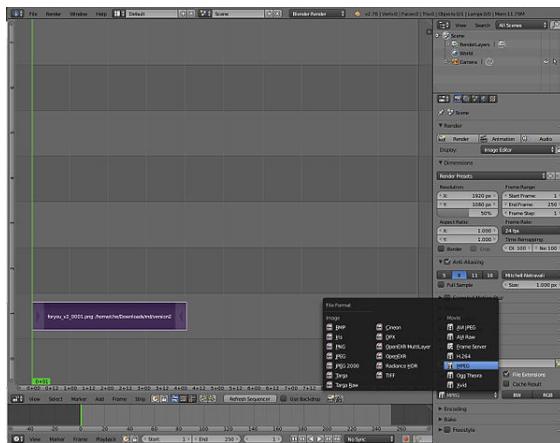
On the bottom of the sequencer window click Add>>Image then browse to the location where you saved the images, Press A to select all the images and finally press "Add Image Strip" on the top-right.

- Set final movie format.
- Set the location that the movie will be saved to by changing the output path.
- Change the End Frame.
- Render!

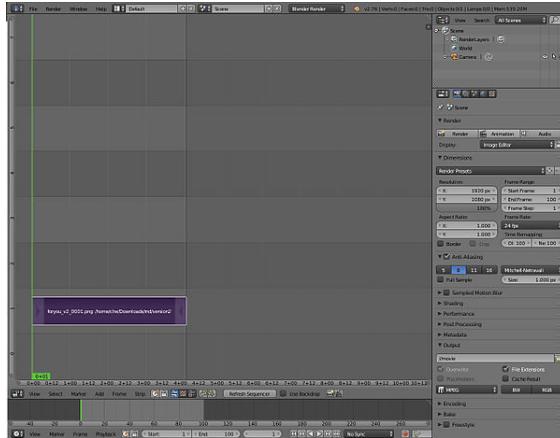
Just Press the Render "Animation" button and wait a few seconds till it's done. Browse to the location you saved your movie and enjoy the animation.



video Sequence Editor



Format



Ready for Render

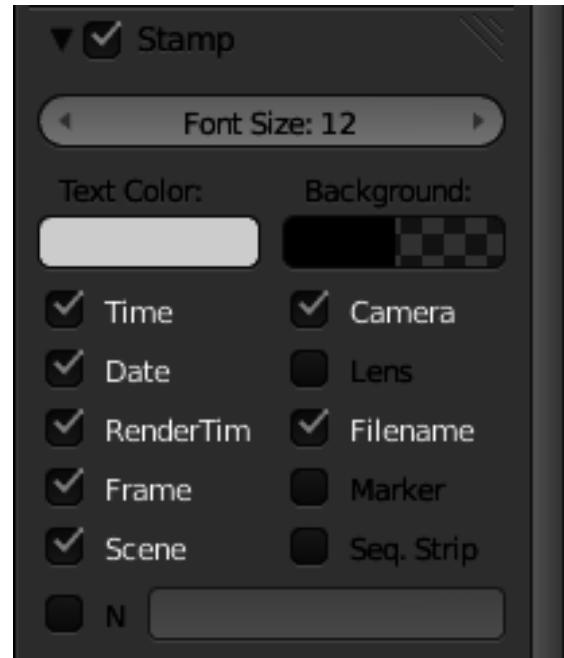
3.21 Further Rendering Options

In previous units on renderer settings, you learned about

- Basic settings, and
- Animation rendering.

Here we will describe some more renderer settings that can be useful in certain circumstances.

3.21.1 Stamping

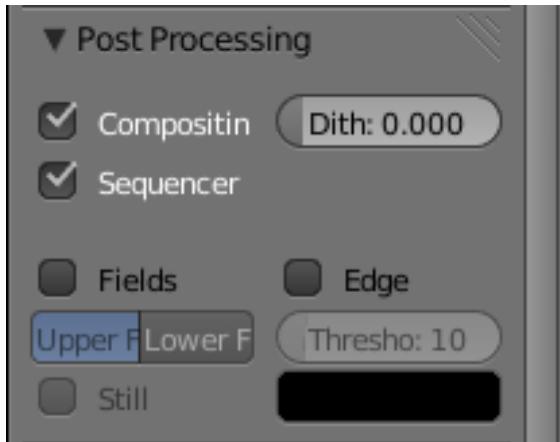


The “Stamp” panel in the Render context provides options for placing descriptive text on top of each rendered frame. The topmost checkbox enables/disables stamping (off by default), while the other checkboxes control the precise information to include, and you can even specify the text and background colour and font size. The information is inserted at the corners of the image, so it can be included in production renders that get appropriately cropped as part of the post-production process. **Note:** In blender 2.76, you'll find it under Metadata. You'll have to check Stamp Output.

3.21.2 Toon Renders

Blender has long had an option for doing simplified “toon” (cartoon-style outlined) renders. In the Render context, look for the “Post Processing” panel and check the “Edge” box. You will also have to set up the lighting and materials for your objects to give a more flat-shaded look.

A more powerful set of options for doing this sort of thing is available with Freestyle, which you will learn about later.



3.21.3 Clay Renders

There is sometimes a need to render a scene without detailed materials and textures, just to see what the objects look like. In the Render Layers context, there is an option to override all lighting and materials for objects in the layer with a particular light group and a particular material. It is common to use a plain diffuse, colourless grey material for this purpose, making all the objects look like they are made out of clay, hence the name.

3.21.4 Transparent Backgrounds

Blender by default provides a “sky” or background for your rendered scene; settings for this can be controlled in the World context.

Sometimes you don’t want such a background at all; you simply want the objects in your scene set against a *transparent* background. So, for example, if you insert the image in a Web page, the scene is displayed against the page background, rather than its own image background.

The way to do this is quite simple:

- Choose to render the Sky as Transparent. In the Render context, look for the Shading panel, where there is a popup menu labelled “Alpha:”, with items “Sky” and “Transparent”. Change this from its default “Sky” to “Transparent”. You have to do this before rendering the image.
- When saving the rendered image, choose a file format (in the Output panel in the Render context) that includes an *alpha channel*; for example, PNG allows for this, but JPEG does not. With such a format chosen, you further have to remember to select the “RGBA” button, not “RGB”, otherwise the transparent areas will simply be filled with black.

3.22 Overview

3.22.1 Introduction

Particle systems are used to simulate large amounts of small moving objects, creating phenomena of higher order like fire, dust, clouds, smoke, or fur, grass and other strand based objects. You may also use other objects as a visualization of particles.

Before you start with the tutorials, you should at least take a brief overview about the very extensive documentation pages of the particle system. You will find every single parameter explained in the manual if you have the desire to delve deeper ...

Don’t forget: particles alone don’t do any magic. They are only a placeholder for something nice to view. You have to take care of the visualization also, and that is usually the harder part than to create the particle system.

3.22.2 The very first particle system

Creating a particle system

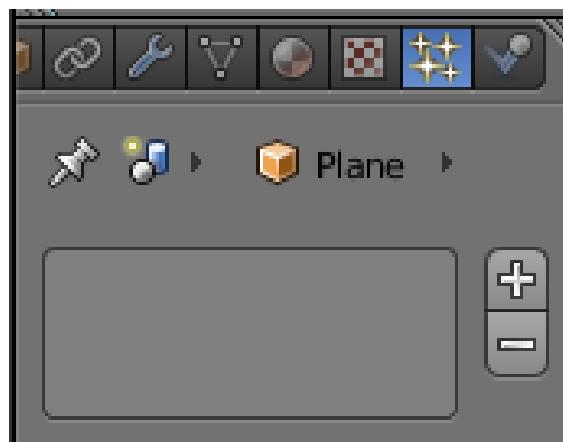


Image 1a: Where to create a new particle system

To create a particle system:

- Create a mesh object to be the “parent” (source) of the particles (only mesh objects can emit particles). Let’s use a simple plane to start with.
- select the object
- change to the Particles tab in the Object Properties window
- click on the “+” button (**Image. 1a**)

Voila, your first particle system (**Image. 1b**)! It doesn’t do anything useful now, but we’re going to change that on the following pages.

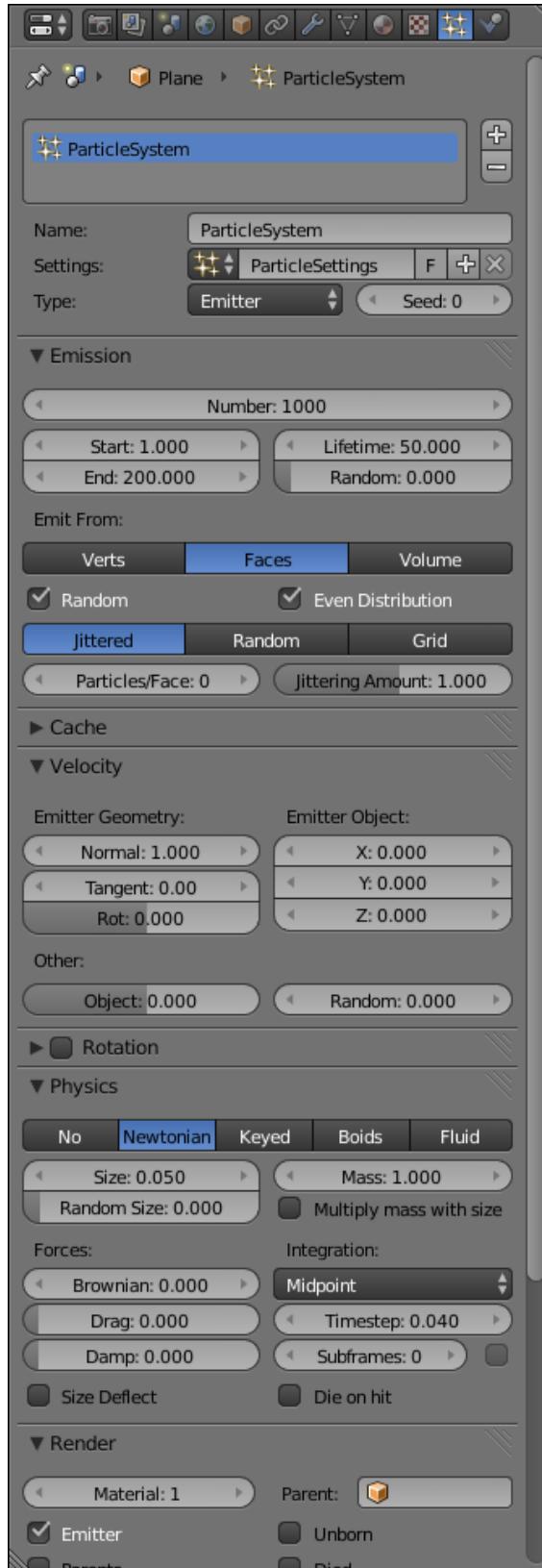


Image 1b: The very first particle system

- To see the particles, you must start the animation running by pressing Alt + A . You will see particles appear and fall from the plane. Press Esc to stop

the animation; you return to frame 1. The particle movement is cached (stored), so if you play your animation again it will go faster (well, you won't notice any difference in this simple case).

- If you want to stop the animation in the current frame, press Alt + A while it is running instead of Esc .
- The shortcut for returning to the first frame is Shift + ←)
- To see the particles even better change to wireframe mode (Z),

If you change anything in your particle system you always have to return to frame 1, to recalculate the system from the start.

Use the *timeline* window along the bottom of the screen to change easily between frames.

Changing properties of the system

Some important settings, from the “Emission” panel:

- Number:* the total number of particles; increase this to 5000
- Start:* and *End:* the start and end frame of the emission
- Lifetime:* the lifetime in frames of the particles

And in the “Velocity” panel are settings that combine to determine the initial velocity of the particles:

- Normal:* a velocity component in the direction of the face normal (if emitted from faces)
- Tangent:* a velocity component parallel to the face
- Rotation:* controls the direction of the tangent velocity component
- Emitter Object X/Y/Z:* a velocity component oriented in the object’s coordinate system
- Object:* a multiplier that imparts some proportion of the object velocity to the particles (try moving the object around with a nonzero value for this field to see its effect)
- Random:* a random contribution to the object velocity

Initially, the plane has its face normal oriented upwards. However, it probably looks like the particles are emitted downwards. This is because the initial normal velocity of 1.0 is quite small compared to the force of gravity (which is on by default). Try increasing it to something like 10.0,

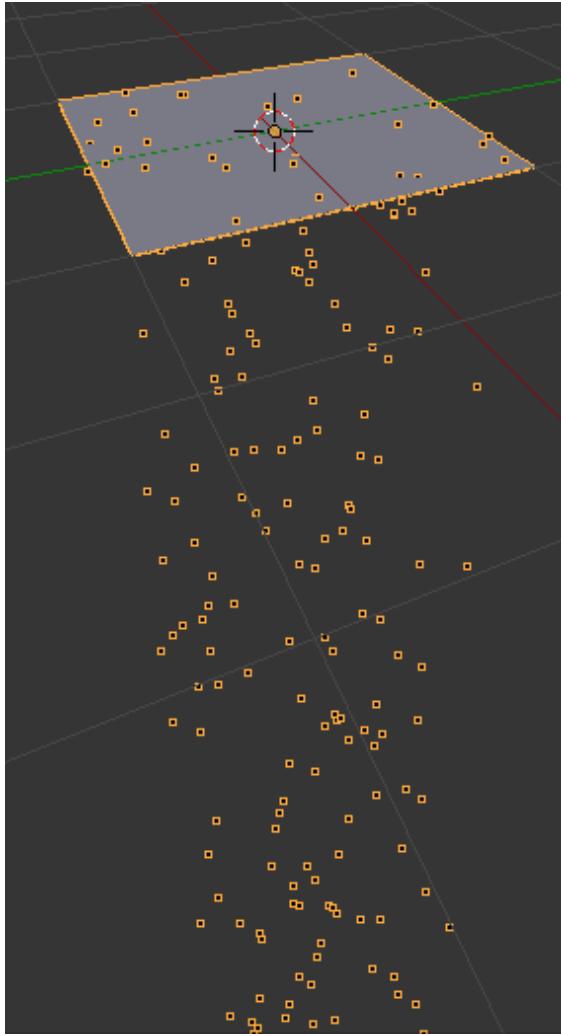


Image 2a: Particles emitted in the direction of the face normal of the plane

and when you rerun the animation, you should see the particles rise quite high above the plane before falling down again.

If you render a frame with particles showing, you will see the particles appear as white blobs. This is the default *Halo* rendering of the particles.

Changing the material of the particles

- Switch the Properties window to the Materials tab and create a new material for the plane.
- Change the material type to *Halo*. (see also the [Manual on Halos](#)). Halos are a post rendering effect, that is applied after the scene is finished. So halos can't shed any lights on other objects, they are not rendered behind *RayTransp* materials (like glass).
- Set the color to deep blue (RGB: 0/0/1)

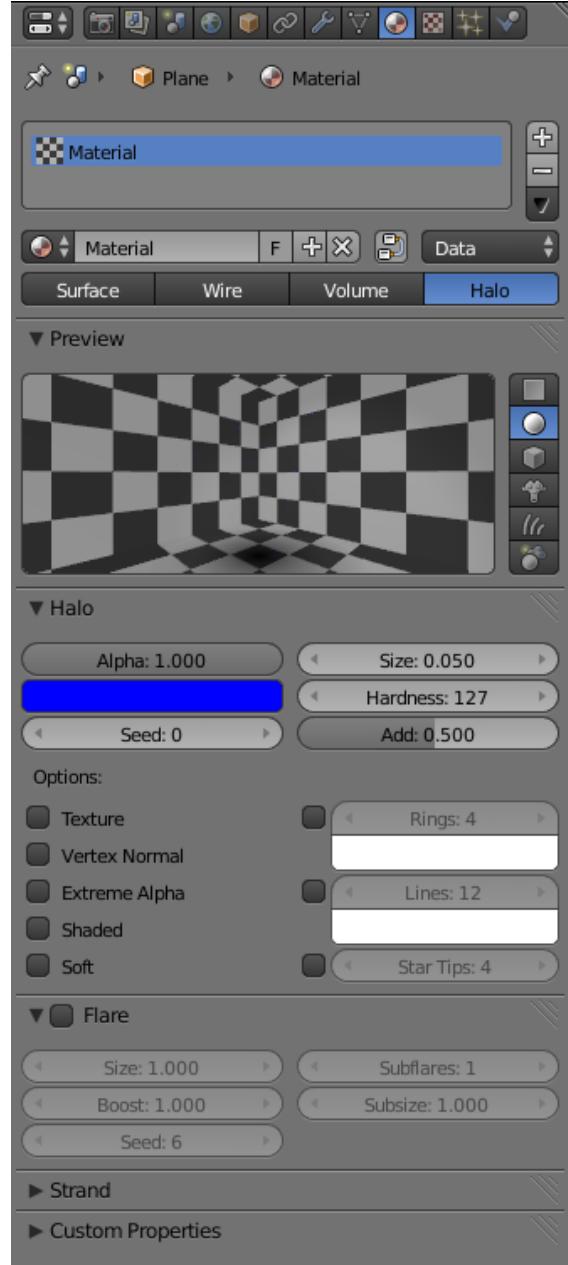


Image 3a: The first simple Halo material.

- Size:* 0.05 so each halo is quite small.
- Hardness:* 127 so that each halo has the maximum sharp edge
- Add:* 0.5 so that the brightness increases where several halos overlap

Set the world color to black and render (**Img. 3b**). Nothing special till now, but that will change soon. So proceed to the next page, where we're going to make some fire.

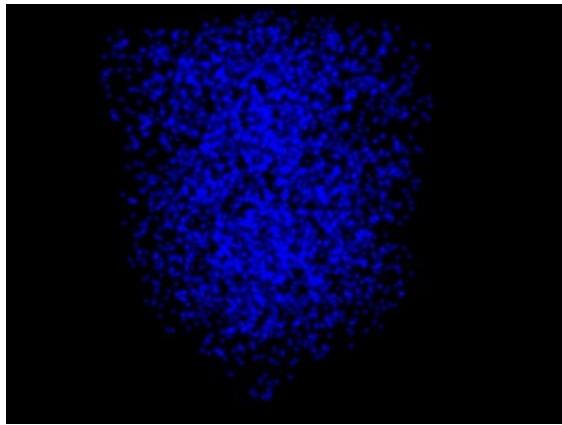


Image 3b: Our first particle system rendered in frame 68

3.23 Fire

We're going to create a camp fire with a simple particle system. This tutorial is based on the method described in the [Blender Manual](#). The result of this tutorial is shown in **Fig. 1**, the Blend-File is included at the bottom of this page.

If you need more realistic looking fire, you should use the method described in [BlenderArt Magazine No. 16](#), though that method is more advanced and uses Compositing Nodes heavily.

The starting point of the tutorial is how fire behaves physically. The flames are made of hot gases. These accelerate upwards due to their lower density in contrast to the cooler air in the environment. Flames are in the middle hot and bright, to the outside they are darker.

3.23.1 The particle system

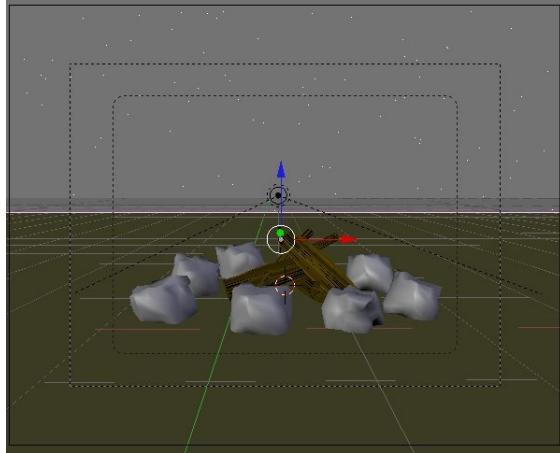


Figure 2a: A simple scene

I've created the usual scene with some stones and a few pieces of wood (the wood is by courtesy of Teeth). (**Fig. 2a**).

- Add a *Plane* in the middle of the stones.

This will become the particle emitter.

- Rename the *Plane* object to “Emitter”.

If you use good names you will find it much easier to orientate yourself in your scene later. Having 100 objects named “Cube.something” will make it very difficult to quickly select a desired object.

- Subdivide the plane once in *Edit mode*.
- Change the shape of the plane, so that its shape equals the base of the fire.
- Change to *object mode*.

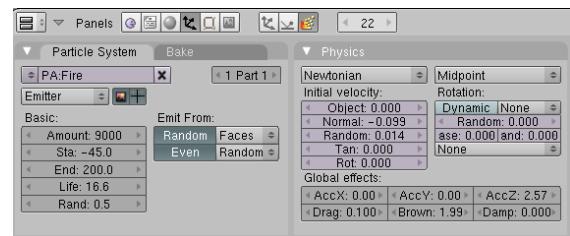


Figure 2b: The particle system

- Change to the *Particle* buttons of the *Object* buttons.
- Create a particle systems (*Add New* in the *Particle System* panel - and make sure the plane is selected!).
- *Type: Emitter* The plane emits the particles.
 - *Amount:* 9000 The total amount of particles.
 - *Sta:* -45 The simulation shall start before the rendering, to have a fully developed flame in the first frame.
 - *End:* 200 The simulation shall last 200 frames - here: the particles are emitted till frame 200.
 - *Life:* 16.6 I've adjusted the lifetime of the particles to their speed. Both parameters together regulate the height of the flame.
 - *Rand:* 0.5 The lifetime is changed randomly.
- *Emit from:*
 - *Random*
 - *Faces*
 - *Even*
 - *Random*

This creates particles with a random distribution on the faces of the emitter object.

The movement of the particles is controlled with particle physics. You set the *Initial Velocity* and let the physics do the rest.

- *Normal*: -0.099 The particles are emitted slightly against the direction of the face normal. This leads to a bit wider fire at the base.
- *Random*: 0.014 This creates a random start velocity as well in speed as in direction (you could use a texture to randomize only the speed, see the discussion page for that).

After you have given the particles an initial velocity they are moved by forces.

- *AccZ*: 2.57 A force in positive Z direction (upwards).
- *Drag*: 0.1 Air drag decelerates the particles.
- *Brown*: 1.99 Random movement simulates agitated air movement.



Figure 2c: Particles without material

The particle system is finished. Until now it doesn't look like much (see the white Blob in **Fig. 2c**). Therefore the emitter will get a material, this material will be animated.

3.23.2 Material

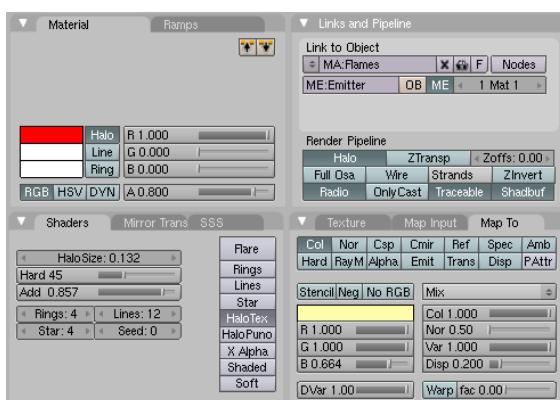


Figure 3a: Material for the emitter object.

- Create a new material for the emitter, call the material *Flames*.
- Activate the *Halo* button in the *Links and Pipeline* panel. Else we couldn't set the particle parameters. The particles would be rendered with the default Halo values.
- *Halo color*: 1/0/0 (red)
- *Alpha*: 0.8 The particles shall always be a bit transparent.
- *HaloSize*: 0.132 I wanted many, but fairly small particles.
- *Hard*: 45 The transition from fully transparent to fully opaque.
- *Add*: 0.875 Several *Halos* over each other combine their power. This makes the fire in the center really bright.
- *HaloTex*: A Halo can bear an individual texture, but only the texture in the first texture slot is evaluated.

To give the Halo a bit more structure, give it a texture:

- Add a new texture in the first texture slot.
- *Map To*:
 - *Col*
 - Color: Bright yellow (1/1/0.664)

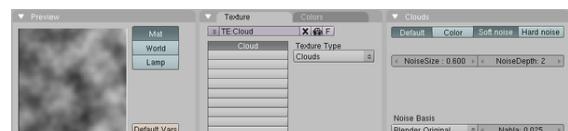


Figure 3b: Adjusted texture

- Use an adjusted *Clouds* texture with a *NoiseSize* of 0.6.

Animation of the particle material

The particles “pop” into life and vanish suddenly. We should change that. Therefore we're going to animate the *Alpha* value of the particles.

- Make sure the material buttons are visible in the buttons window.
- Change to frame 21, move the mouse cursor over the button window and press **I->Alpha**. This is going to be the maximum visibility of the particles.
- Change to frame 1. Change the *Alpha* value to 0 and insert the next key.

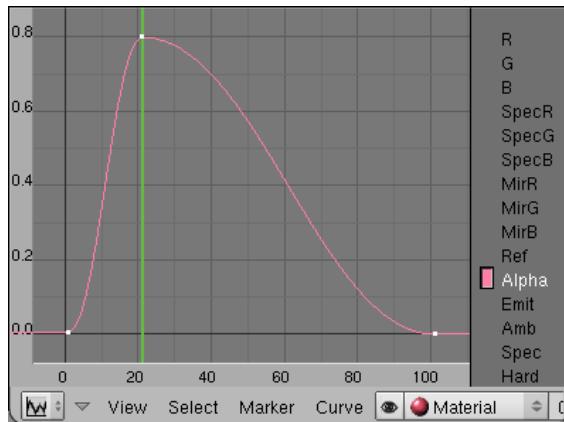


Figure 3c: Animation of the Alpha value

- Change to frame 100. Change the *Alpha* value to 0 again and insert the third key.

If you want to see the IPO curve in the *IPO Editor* window you must change the *IPO Type* selector in the window header from *Object* to *Material*.

The *Alpha* value therefore changes during the individual lifetime of each particle from 0 to 0.8 and back to 0 (**Fig. 3c**).

3.23.3 Rendering

Our particle animation is finished.

- Change the end frame in the *Anim* panel of the rendering Buttons to 200 and click on *Anim*.

Note: If after rendering your particles are too small, such that the fire doesn't look realistic, try increasing the *Halo size* slightly. I used 0.300 instead of 0.132

To actually let the fire glow you have to use one or more lamps and animate them as well. But that would be part of another tutorial ...

3.24 Fur

There is an older version of this page created with Blender v2.40.

(NOTE: **New:** marks

Notes added for newer versions. I'm not an expert but they seem to work.)

This tutorial deals with fur, i.e. lots of relatively short hairs covering a body. We will use particles to create the fur, and discuss a few aspects here:

- How to determine the length and the thickness of the hair.



Figure 1: The result of this tutorial: some furry thing

- How to determine the place to grow the hair.
- How to color hair.
- How to render efficiently.

The particle system is far too complex to show more than one method in this tutorial. You can achieve many of the same effects shown here in different ways.

3.24.1 The emitter

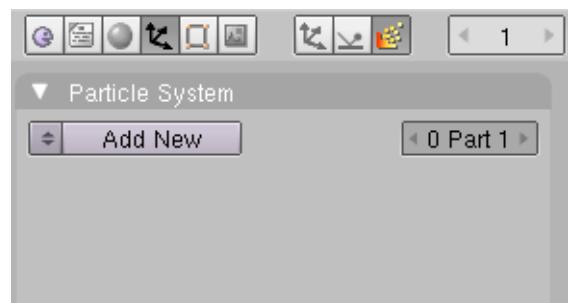


Figure 2a: Adding a particle system.

- Remove the cube.
- Add an *UVSphere*. This will become our emitter.
- Change to the *Particle* buttons in the *Object* buttons (**Fig. 2a**). **New:** looks like in newer versions.
- Click on *Add New*. **New:** Click on in newer versions.
- Rename the particle system to "Fur".
- Change the particle system type to *Hair*.

A *Hair* particle system has a lot of specialties, the most important thing is that we can edit the particle "motion"

by hand if we want to. Apart from that normal particle physics apply, so everything a particle does hair can do also and vice versa. A particle hair shows the way of the particle during its lifetime at once. To do that efficiently not every single frame is rendered as a point, but a certain number of control points are calculated. Between these control points there will be drawn an interpolated path. The number of control points is the number of segments + 1.

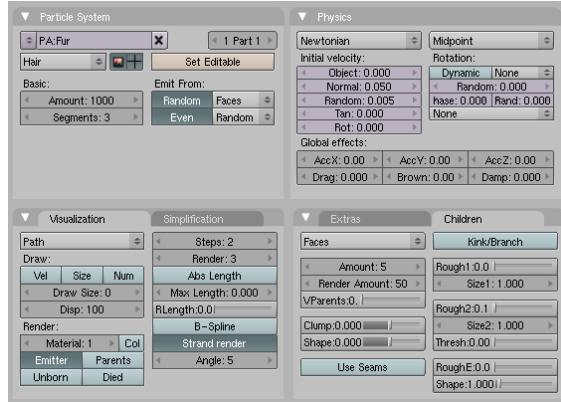


Figure 2b: Particle system settings

For fur you need lots of particles, like 1.000.000 upwards. This will hurt us badly if we have to deal with so many particles in the 3D window and want to render it. Therefore we will create the great amount of particles with so called "children", that mimic the behavior of their parents. The amount of particle parents should be as low as possible, but you need a certain amount to control the distribution of the hair. We will also use as little control points as possible, three segments should be enough for short fur.

- Set the *Amount* to 1000.
- Set the number of *Segments* to three. **New** I can't find this in 2.69. **New2** In 2.7x, it's right next to the particle system type you just chose (emitter or hair), below the slider *Seed*.
- *Emit from:* **New** I can't find this in 2.69. **New2** In 2.7x, you must check the *Advance* box near the *Segments* slider in order to see it.
- *Random:Faces*
- *Even :Random*

This will create a nice, uniform distribution.

Let the hair grow - the hair shows the path of the particle:

- Set *Normal* to 0.05. **New:** Hair Length under Emission. **New2** In 2.7x, you must check the *Advance* box in order to see the *Velocity* panel and adjust the *Normal* and *Random* slider. If you change the *Normal* value, *Hair Length* will set itself to 0.2.

- *Random 0.005*

Nothing special here: the hair grows in the direction of the face normals. Length and direction are a bit randomized.

The *Visualization* type changes automatically to type *Path* if you select a hair particle system. If you would render now, you couldn't see the emitter object any more.

- Activate *Emitter* in the *Visualization* panel. **New:** In the Render panel.
- Activate *Strand Render*. **New:** In the Render panel.

The *Strand Render* (which I have baptized *keypoint strands* to differentiate from the "normal" *polygon strands*) renders the hair strands extremely efficiently and magnitudes faster than the normal strand. It is the only way to handle many hairs in terms of memory consumption. But it has a few disadvantages:

- They are not seen by raytracing, so you don't get raytracing reflections and no raytracing shadows. You can use environment mapping to compute the reflections and *Spot Lamps* with buffer shadows for the shadow.
- If the hair is very thick (like 1 BU) sometimes the shape is not correct.
- Activate *Children from Faces*. **New:** I chose Interpolated under Children. Interpolated is known to give better results when making Fur. Simple, maybe, will work with one color, but we're going to use 2 colors from an image.
- *Amount:* 5 This is the amount of children particles for each parent.
- *Render Amount:* 50 This is the amount of particles during the render.
- *Rough 2: 0.1 New:* I used *Roughness > Random* for this setting. Random variation of the shape of the particles. So the hairs will not stand plain upright and appear a bit curly.

The parent particles are not rendered by default, so now we have 5.000 Particles that render on my old machine in 6 seconds. If we use 1.000 children we have 1.000.000 particles, that need approx. 1 GB of RAM and render in 1:42 minutes. If you render *keypoint strands* with *Children from Faces* you can also use *Child simplification*, which will reduce the amount of particles on objects far away from the camera automatically. **New:** Don't see this when using Children - Simple, When using Children - Interpolated, you'll see it all down the Render section. However, the result is very bad, with the standard settings.

Now we should change the lighting to get a preview.

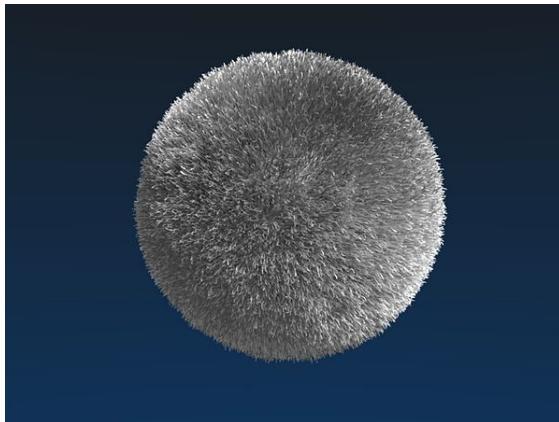
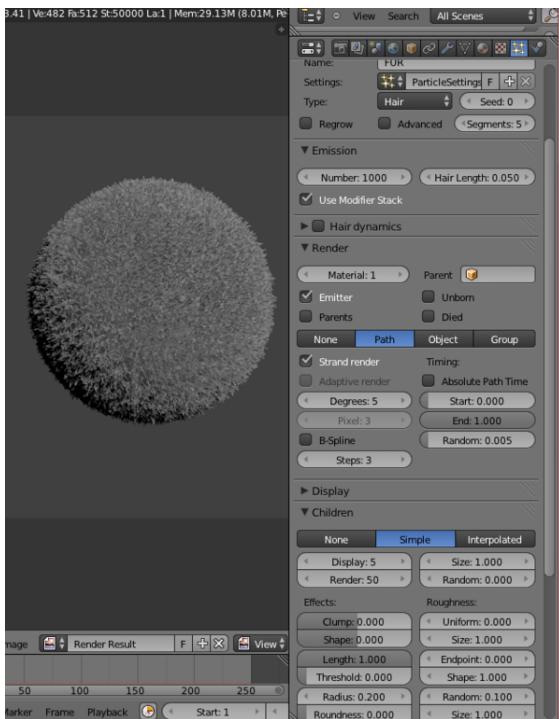


Figure 2c: The first render without material.

- Select the lamp.
- Change to the *Lamp* buttons.
- Change the lamp type to *Spot*.
- Change the shadow type to *Buf. Shadow*
- Change the buffered shadow type to *Classic-Halfway*.

This is a great shadow type that renders keypoint strands very well and creates fewer artifacts than *Classical* (In my opinion). I have inserted two other lamps and used a classical three point lighting for the first rendering (**Fig. 2c**).



Click on image to see larger version.

NEW:

Here are the settings I used and the render result I got. (Click to enlarge the image and read the settings.)

3.24.2 Material

In the material buttons you can set different aspects for the strands:

- their width and form
 - the used shader
 - the base color
 - a texture along the strand
 - different particle attributes like length, density or roughness
- Add a material to the emitter.
 - Name the material “Fur”.

Strands Shader

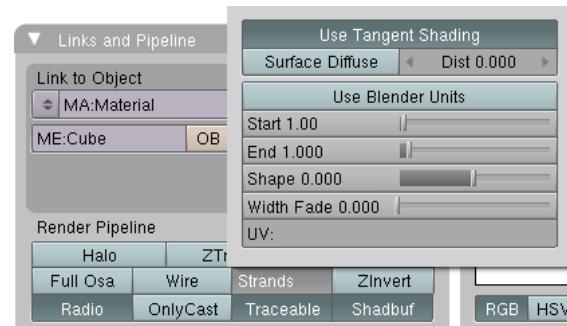


Figure 3a: Strands settings in the material buttons.

The default strands settings for *Keypoint strands* are shown in **Fig. 3a**. Take a look in the Manual about Strands for an explanation of all settings.

- Change the *End* value to 0.25, this will make the hair more spiky (not shown in **Fig. 3a**). **New In 2.7x**, it's called *Tip* (and *Start* has become *Root*)

Giving the hair its base color



Figure 3b: Fur color texture

Strands are rendered with the material of the underlying face/vertex, including shading with an UV-Texture. Since you can assign more than one material to each face, each particle system may have its own material and the material of the underlying face can be different from the material of the strands. We will use an UV texture and use it for the surface of the emitter as well as for the color of the hair.

- Change to *Front* view in the 3D window (*View->Front*).
- Make sure you are in *Orthographic* view mode (also in the *View* menu).
- Change to *Edit* mode of the sphere.
- Press **U** to unwrap, select *Sphere from View*. This is a quick and well working method to correctly unwrap a sphere the easy way.

You don't need to assign a texture in the *UV/Image Editor*, we only need the coordinates now.



Figure 3c: Emitter with color texture.

- Add a texture to the material, name it "FurColor".
- Set *Map Input* to *UV*.
- Go to the *Texture Buttons* and set *Texture-Type: Image*.
- Load an image texture. I have used the image in **Fig. 3b**.

Normally I would just stop here, I think the material is good enough. But if you want to make the fur more fluffy and soft, you should a second texture along the strand, which changes the alpha value. **New:** For this to work you have to choose *Interpolated* instead of *Simple* in the *Children* panel of the particle system.

If you want to do that:

- Activate *ZTransp*.



Figure 3d: Settings for a texture along the strand.

- Add a second texture.
- *Map Input: Strand*
- *Map To: Alpha and Spec, DVar=0*
- Use a blend texture (*Linear* or *Quad*)

You can change all other properties this way, for example the color along the strand (bleached tips).

3.24.3 Changing Hair length with a texture

At first I will show you how to render the emitter mesh with a different material than the strands. Then I will show how to change the length of the hair with a texture semi-interactively.

- Change to the *Editing Buttons*.
- Change to *Edit* mode.
- In the *Links and Materials* panel click on *New* in the material section (see **Figure 4a**).

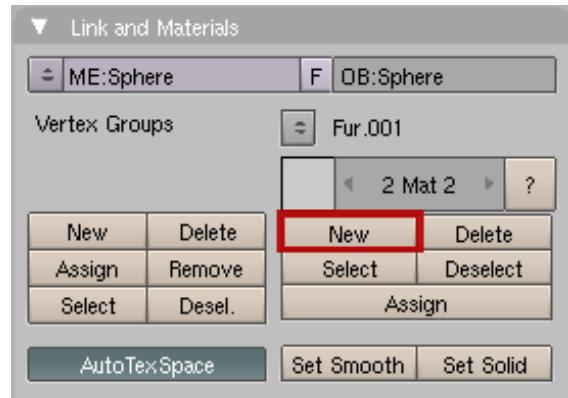


Figure 4a: Button to click

- Make sure all vertices are selected.
- Click on *Assign*.
- Change back to *Object* mode.

Now the emitter bears a second material.

- Return to the material buttons.
- In the *Links and Pipeline* panel click on the **X** next to the material name (*Deletes link to this Datablock*).
- Add a new material.
- Name it *Emitter*.

Now you have a new material on your emitter object. Since the particle system uses material no. 1 you can use different settings for the emitter.

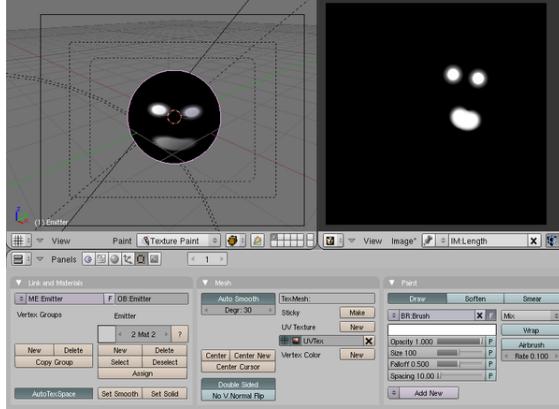


Figure 4b: Texture painting

We have already unwrapped the emitter, this is something that will probably be the case also for any real models. Now we will use an UV-Image and texture painting to determine the hair length.

- Split the 3D window.
- Change the right hand side to an *UV/Image-Editor* window.
- Change to edit mode of the emitter object.
- In the *UV/Image-Editor* use *Image->New...* and confirm the default settings. This will create a new image, that we will paint on.
- Click on the package icon in the windows header of the *UV/Image-Editor* window. Confirm.
- Change the object to *Texture Paint* mode. **Noob Note:** *In the 3D Window, click on the drop down menu. Change it to Texture Paint.*

Now you see the texture on the object.

- Paint a structure on the object.
- Change back to object mode.
- Go to the *Material* buttons.
- Change the active material to *Fur* (click on the arrows in the *Links and Pipeline* buttons where it reads *2 Mat 2*).



Figure 4c: Material settings for setting the particle length with a texture.

- Create another texture. Name it *FurLength*.
- Set *Map Input* to *UV*.
- Set *Map To*
 - Turn *Col* off.
 - *PAttr*
 - *Length*
 - *DVar=0* All the white areas on the texture will produce a particle length of 0.
- Load the image texture that we have painted.



Figure 4d: Controlling particles with a texture: result

The result is shown in **Fig. 4d**, you can also see the particles change in the 3D window.

New In Blender 2.7x, things have changed. I followed until the step *Change the active material to Fur*. Then, I went to the *Texture* tab and clicked on the *Particles Texture* button next to the *Material Texture* button (or you can click on the *Particles* tab button and then click on the *Texture* tab button). I added a new texture, opened the one I had just painted in the *UV/Image-Editor*, set *Coordinates* to *UV* (in the *Mapping* panel). Finally, I enabled *Density* in the *Influence* panel and set its value to *-1.00* (*1.00* has the opposite effect = hair on the white areas of the texture). The result should be the same as above.

There would have been other ways to achieve this result, e.g. with vertex groups or with particle editing. But I like to work with textures, because you have very fine control and may change the strength of the effect at any time. Vertex groups don't allow for such fine control or you need

very many vertices in the emitter. Particle editing (what we will do in the next step) is lost if you change the base particle settings late on, and you can't change its effect so easily.

3.24.4 Comb it!

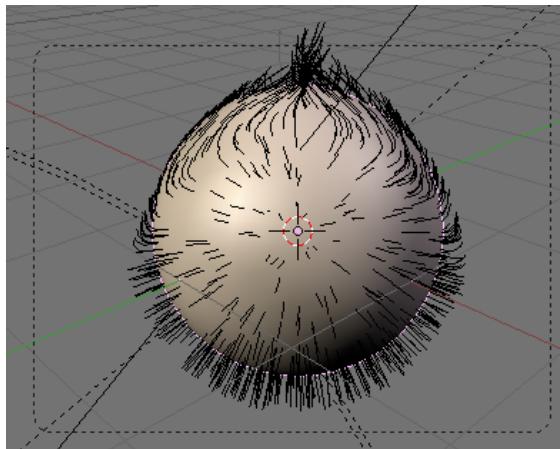


Figure 5a: Combing in Particle Mode

An effect that is often underestimated is the importance to comb fur in the natural directions. Fur doesn't simply stand upright, and it also doesn't follow gravity (or only to a small amount). So back to the particle system!

- Change back to the *Particle buttons*.
- Click on *Set Editable* in the *Particle System* panel.
- Change to *Particle Mode*.

Particle Mode only appears if you have made the particle system editable, and only hair systems can be made editable. There are a few lifesavers to know when working in *Particle Mode*.

- you can edit only the control points (remember the setting *Segments* from the beginning of this tutorial)
- you can only edit parent particles, so you need enough parents for good control
- Activate *Limit selection to visible*
- Activate *Point select mode* for even finer control.

Both settings are in the window header of the 3D window.

- Open the *Particle Edit Properties* panel with **N** key in the 3D window.
- Select *Comb*.

You have quite a few different tools at hand in *Particle Mode*, see the [manual on Particle Mode](#).

- Comb the hair following the natural flow.

On the example sphere I have used here it is a bit difficult to tell what the natural flow should be ;-. So I have just very carefully combed and changed the length at a few places a bit. You find the rendered result in **Fig. 1**, the Blend file is linked below.

3.24.5 Links

- A tutorial in German that shows how to create Grass with a very similar method.
- Awesome fur shown here: [Tiger](#)

3.25 Fireworks

[Note: In newer versions of Blender, the reactor particles are gone.]

[if you use 2.6x version of blender, then you could watch the YouTube video {particles from particles part1/2}. It's a great tutorial, but the part 2 of it uses compositing, which is described in this book a little later. Therefore, you could apply the knowledge you have learnt from this book to animate the explosions!]

How to create a firework from particles? We will use cascaded particle systems especially of the type *Reactor*. Abstract:

- We create a emitter object in the appropriate size.
- Then we use three successive particle systems:
 - the first of the type *Emitter*
 - the second of the type *Reactor*. This system reacts to the death of the emitter particles.
 - the third again of the type *Reactor*. This systems reacts to the proximity of the second systems and thus creates a "drag".
- We create three different materials (one of them animated) and assign them to the three particle systems.

3.25.1 The Emitter

We use a *Plane* as emitter object. Scale it to your liking. I have used a relatively large particle system, so my plane has a length of 15 BU (Blender units).

- Add a particle system (**Fig 2a**, left hand side).

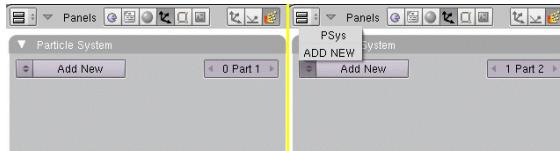


Figure 2a: Adding particle systems. First particle system on the left hand side, a second particle system on the right hand side

- *Amount:* 25
- *Sta:* 1
- *End:* 250
- *Emit from:*
 - Random
 - Faces
 - Even
 - Random
- *Initial Velocity*
 - Normal: 22
 - Random: 7
- *AccZ:* -9.8
- *Visualisation: Line*
 - Back: 1.1

A few particles (25) are created in the first 250 frames of the animation and emitted upwards. Gravitation, velocity and lifetime are adjusted so, that the particles reach the end of their lifetime at the topmost point of their trajectory.

The *Line* visualization lets the particles appear as long drawn-out line.

- *Bake* the particle system. Use 500 as the *End* frame for bake. 300 would be sufficient here, but 500 do no harm.

Noob Note: Bake is in the Bake tab next to particle system tab in particle buttons window.

The plane will get a *Halo* material.

- Activate *Halo* in the *Links and Pipeline* panel.
- *Halo:* color red
- *Halo Size:* 0.421
- *Hard:* 35

I've adjusted the size and hardness of the halo so long until I liked it, so there is no specific reason to use these values.

3.25.2 Reactor 1

- Click on the arrow next to *1 Part 1* in the *Particle System* panel.
- The settings for the first particle system are no longer shown, now it reads *1 Part 2* (**Fig. 2a**, right hand side).
- Click on *Add New*, now it reads *2 Part 2*.

You just have created a second particle system. Rename this system, a good naming convention will help you a lot to keep the overview.

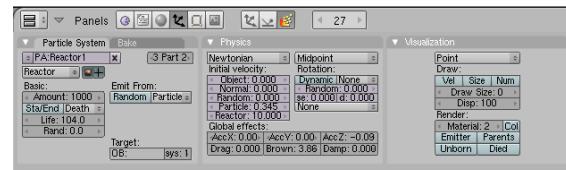


Figure 3a: The first reactor system

- Activate *React on: Death*.
- *Life:* 104
- *Emit from: Particles*

You don't have to set anything for the *Target*. It is not necessary to set the target object if it is the same object as the reactor. You just have to set the number of the target particle system eventually.

The *Reactor* particles react on the *Death* of the particles of the target system. They will be emitted from the point that the particles occupy at their death.

To control the movement of the system, I have made following setting in the *Physics* panel:

- *Particle:* 0.345
- *Reactor:* 10
- *AccZ:* -0.09
- *Brown:* 3.86

Because of the *Reactor* setting the star is moving away from the emitting particles. The *Brown* movement lets the trajectory appear trembling like affected by wind.

- Set the material number to "2" in the *Visualization* panel.
- Bake the system, again use 500 as end frame.



Figure 3b: Material settings for the 1. Reactor system

Material 2

- At first you have to assign a new material in the *Link and Materials* panel of the *Editing Buttons*.
- Use the material settings from **Fig. 3b**,
- The material also gets a *Clouds* texture with *Noise depth* “3”, simply leave all other settings unchanged.

The texture is actually superfluous here, but the next particle system is given the same texture, and colors should be adjusted a bit.

- Animate the *Alpha* value of the material.
- Set the first Ipo key in frame 1 (*Alpha=1*). To do that move your mouse cursor over the *Buttons* window. Press **I->Alpha** (and change ipo type to material in the ipo curve editor if you are using that).
- Set the second Ipo key in frame 86 (*Alpha=0.75*).
- Set the third key in Frame frame 101 (*Alpha=0.0*).

The second particle system is faded off relatively quickly, but the particles don't disappear suddenly.

3.25.3 Smoke Trail

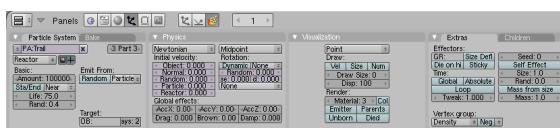


Figure 4a: Particle system for the trail

The third (and last) particle system is again a *Reactor* system and is reacting to the second system. It's going to react to the *Nearness* of the particles. Without moving the variation of the particles is created with a texture and a random variation of their lifespan. What Blender considers as close can be changed with the particle *Size*.

- Create a third particle system (like you created the second).
- Use the settings from **Fig. 4a**. Important are the settings:

- *Amount:* 100000

- *React on:* Near
- *Emit from:* Particles
- *Life:* 75
- *Rand:* 0.4
- *Target Sys:* 2
- *Material:* 3

- Bake the system, again use 500 as end frame.

Material 3

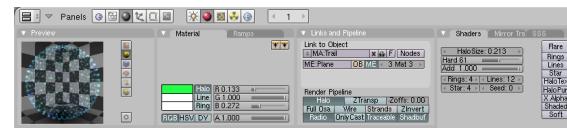


Figure 4b: Material for the Trail particle system

- At first you have to assign a new material (the third) to the object in the *Link and Materials* panel of the object buttons.
- Assign the existing *Clouds* texture from material 2 also to material 3, but select a dark blue as target color (in the *Map To* panel).

3.25.4 Render

- Render your animation, use end frame 500.

That's all. I hope you were able to follow the tutorial. Criticism and suggestions, write best on the talk page or improve the article directly.

3.26 Particles forming Shapes

Note all the blender users of version 2.6X, 2.7X: Using sort of a domain to dissolve the particles by using the alpha parameter of the material is how I did it. You should understand along the way. I'll write the sentences in key sequences to reduce the size. So [s], 0.5, [enter] means to scale the object by 0.5. Don't worry you'll learn keyed physics here.

- Open a new blender file.
- Delete the default cube.
- [shift]+[A],text.
- [num0]
- In the menu, below the tool shelf check align to view.

- Select the text object, [alt]+[c], convert to mesh.
- [shift]+[a], add Suzanne.
- [ctrl]+[2], not the numpad.
- Set smooth.
- In the modifier's tab apply the subsurf immediately.
- If you want you can scale the objects.
- Add a particle system to text object.[I leave the renaming parts to you]
- In the emission subpanel of the text object's particle system:

Number:5000; Start/end(both):1; Emit from: faces, check random and even distribution, random; Physics: none; Uncheck emitter in the render sub-panel and choose none rather than halo.; Set all to zero in the field weight's sub panel(actually I didn't notice much difference it even being on).;

- Now add a particle system to Suzanne and give it the same particle system but make sure to have it made to a single user(just press the '2' beside the system name).
- [shift]+[a], uv sphere.
- Selecting the sphere, [alt]+[g](just in case).
- [s],2000,[enter].
- [ctrl]+[2],set smooth and apply the modifier.
- Give the same particle system to the sphere.
- Now [num0](assuming you're not already in camera view).
- [LMB] on the pass partout(the greyed out area except for the camera).
- [shift]+[a], add cube.
- Give the cube a particle system.
- Assign the same particle system but let's make a few changes:

In the physics subpanel select keyed.; In the render subpanel choose halo.; In the menu labeled keys, click the '+' icon thrice and set the target object to text , Suzanne and sphere in order.; Now select the Suzanne's particle system in the keys menu, check use timing and set it to 200. Now select the sphere's and set the time to 7025! (well this is just to lower the speed we won't render this); And set the life time of these particles to 500.(this is in the emission subpanel);

- In the timeline header set the end frame to 300.

- Now for the material. Select the cube and:

Give it a new material.; Select halo.; Size: 0.03; Hardness: 16; Add: 1; RGB: As you desire how the particles appear.(I did RGB=0, 0.106, 0.8); Now go to frame 260 and insert a keyframe to alpha value.; Now go to frame 270 and insert make the alpha value zero and insert another keyframe.;

- Set the scene's gravity off.
- [ctrl]+[F12](I suggest you render this one in xvid rather than png it's fun)

I will show on this page two ways to let particles take the shape of other objects.

1. keyed particle physics (**Img. 1a**)

2. a harmonic force field with a damping of 1. The force field is emitted by another particle system, so that each particle is attracted by another particle (**Img. 1b**).

v2.48 The second method is easier to control, but I've found that it may not work exact enough. Even if you use RK4 as calculation method, there are some glitches in the precision of the calculation, so that your particles may not come to a standstill (depending on the size of the particles and their number this may be visible or not).

v2.49 That the particles didn't come to a standstill has been a bug that Jahka has fixed in 2.49. So if you use Version 2.49 and upwards you will find it easier to work with the harmonic force field.

3.26.1 Keyed Physics

So here we're going to use keyed particle physics and quite a few different particle systems.

Basically this is the way to go:

- Use mesh objects and let them emit particles.
- Each system moves to another system because of the keyed physics.
- Choose a good visualization of the particles and animate that also.
- At last the hard part: sync the different animations.

The animation shall last 300 frames, this will be 12 seconds.

- In the first 2 seconds the word "Blender" shall appear (frame 1 to 50).

- Than Suzanne shall build up in 3 seconds (frame 51 to 150).
- Suzanne shall be clearly visible for a few frames (frame 151 to 200).
- After that the form shall dissolve and the particles move a bit until they vanish (frame 200 to 300)

Preparing the scene

- Remove the cube.
- Add a text object *Blender*. Name it *Blender-first*.
- Add a *Suzanne* (*Add->Mesh->Monkey*). Name it *Suzanne-first*.
- Convert the text object to a meshobject, because only mesh objects can emit particles (menu: *Object->Convert Object Type->Mesh*).
- Move the objects to their final positions, also the camera.
- **Clarification required - what are the final positions?** (**Noob note:** the author probably means to move the camera and objects to a right position for viewing the animation correctly.)

We need every visible object twice.

- Duplicate *Blender-first* and call the duplicate *Blender-last*.
- Duplicate *Suzanne-first* and call the duplicate *Suzanne-last*.

One problem is that the objects have different sizes. So how many particles shall be used to create a clearly visible form? Since the most difficult object is Suzanne, I have first chosen an acceptable representation and set everything else accordingly. But I will discuss the settings of the objects in order, and not in the way I did it in reality.

Set up the particle systems

Blender-first:

- Add a particle system to *Blender-first*.
 - Amount 20.000
 - Sta: 1, End: 50, Life: 50
- *Emit from:*
 - *Random/Faces/Even/Random*

Leave all other settings at default. This object will appear in the first 50 frames. After that the object has to vanish in the 51th frame.

- Insert a layer key for *Blender-first* in frame 50 (mouse over the 3D window, **I->Layer**).
- Change to frame 51.
- Move the object to layer 20 (**M->select layer, bottom right box**). Make that layer visible.
- Insert the next layer key for the object.
- Make layer 20 invisible again (select layer 1).
- Change back to frame 1.

Suzanne-first

- Add a particle system.
 - Amount 20.000
 - Sta: 50, End: 100, Life: 200 (we don't need this long life but the lifetime doesn't matter as long as it's long enough)
- *Emit from:*
 - *Random/Faces/Even/Random*
- *Visualization: None*

Leave all other settings as default. These particles shall not move and they shall not be visible. They are simply a target to the next system.

- Animate the layer of this object also. Move it out of sight in frame 151.

Blender-last

These particles shall move from the shape “Blender” to the “Suzanne” shape.

- Add a particle system to *Blender-last*.
 - Amount 20.000
 - Sta: 51, End: 51, Life: 100
- *Emit from:*
 - *Random/Faces/Even/Random*
- *Physics:*
 - *First/Keyed* It is the first system of a chain (a very short chain in this example) of systems.
 - *Keyed Target: Suzanne-first, Psys: 1*. Target is the first particle system of “Suzanne-first”.

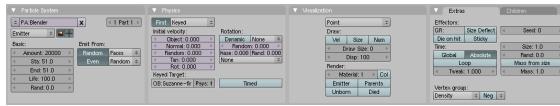


Image 2a: Particle system of the first keyed system.

- Set time to *Absolute* in the *Extras* panel. We're going to animate the material of the particles, I want all particles to change at once in a certain frame.

Suzanne-last

These particles shall dissolve the form, so at first they take the shape and move around randomly afterwards.

- Add a particle system to *Suzanne-last*.
 - Amount 20.000
 - Start: 151, End: 151, Life: 150, Rand: 0.5. The lifetime is randomized from 151/2 to 151 frames. [Noob Note: In 2.49a, you can't set the start later than the end, so you must first change the end to 151, then change the start.]
- Emit from:
 - Random/Faces/Even/Random
- Bake: End frame 300. Though we don't use baking here, you have to set the End frame in the *Bake* panel. Else the last 50 frames wouldn't be calculated.
- Time: Absolute

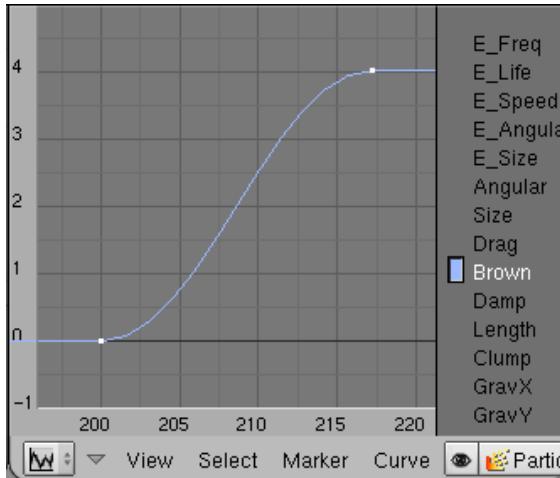


Image 2b: Animation of Brownian Motion

Animate the Brownian motion of the particles.

- Open an *Ipo* window.
- Select *Ipo Type* particles.

- Select the *Brown* channel (**LMB**).
- Ctrl-Click** with the left mouse button in the *Ipo* window to create an *Ipo-key* for the selected channel.
- Create a curve similar to **Img. 2b**. **Brown = 0** to frame 200, than increase to **Brown = 4** in frame 217.

Material

Blender-first and Blender-last will get the same material, Suzanne-first is invisible anyway, but Suzanne-last will get a material. The material of the three objects is very similar, but I have animated the visibility of the particles to match the brightness of the objects.

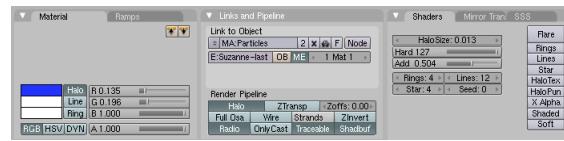


Image 2c: Material settings for the particles

Suzanne-last

- Create a simple *Halo* material like that in **Img. 2c**.

This is nothing special, relatively small particles with a relatively “sharp” edge. The *Add* parameter gives more brightness if particles overlap, I think this is a nice effect here.

Blender-first

- Assign the material also to “Blender-first”.
- Make the material a *Single User Copy* by clicking on the small number next to the material name.
- Set *Alpha* to 0.438.

Blender-last

- Assign the “Blender-first” also to *Blender-last*.
- Make the material a *Single User Copy* by clicking on the small number next to the material name.
- Animate the alpha value of the material.

- Insert an *Ipo* key for *Alpha* in frame 50.
- Change to frame 150.
- Set *Alpha* to 1.
- Insert an *Ipo* key for *Alpha*.

Now the brightness of all systems are matched.

Links

- Manual about Particle physics

3.26.2 Harmonic Force Fields

So here we will use *Harmonic* force fields. The Text “Blender” shall be transformed to the text “2.5” and than to the text “The big leap forward” (**Img. 1b**).

- Use mesh objects and let them emit particles. Only the particles from the first object are visible and move.
- The other particle systems use an animated *Harmonic* force field to attract the particles from the first system.
- The particles from the first system take the place of the other particles.
- Choose a good visualization of the particles and animate that also. This is difficult and maybe you should animate the visualization at last.
- At last the hard part: sync the different animations and force fields.

Preparing the scene

- Remove the cube.
- Add three text objects:
 - “Blender”
 - “2.5”
 - “The big leap forward”
- Convert the text objects to mesh objects (menu: *Object->Convert Object Type->Mesh*).

Set up the particle systems

- Give each of the three objects its own particle system.
- The first object “Blender” gets following system:
 - Type *Emitter*
 - *Amount=4000*. We need a lot of particles, to fill the last text evenly. If you want to make the animation even better, you could animate the *Halo* size of the particles to match the density of the different objects.
 - *Sta(rt)=1, End=10, Life=250*. The particles are created from the first to the 10th frame and live for 250 frames.

Only the particles of this system will be visible later.

- Change the calculation method from *Midpoint* to *RK4*. We need the extra precision here.

The objects “2.5” and “The big leap forward” will get particle systems with very similar settings:

- “2.5”:
 - Type *Emitter*
 - *Amount=4000*
 - *Sta(rt)=30, End=30, Life=70*. So all the particles are emitted in frame 30 and live 70 frames. So their force field (which we give them soon) will start in frame 30 also. If you want a smoother transition you should give the particles a bit more time to be emitted (like from frame 30 to frame 40).
 - Emit from *Random/Faces/Even/Random*.
 - Physics *None*. The target particles shall not move.
 - Visualisation *None*. The target particles are not rendered.
- “The big leap forward”: All settings as for “2.5”, except *Sta, End* and *Life*.
 - *Sta=101, End=101, Life=90*.

The force fields

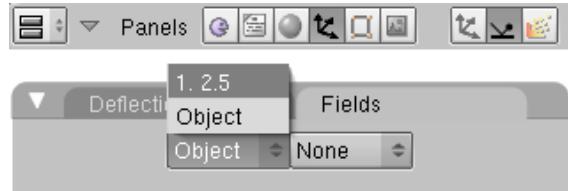


Image 3a: The object does not get a force field.

The **particle systems** of the objects “2.5” and “The big leap forward” will get a force field now. The objects **don't** get a force field.

- Select the object “2.5”.
- Select the particle system in the *Field* panel of the *Physics* buttons (**Img. 3a**).
- Use field type *Harmonic*. Now the particles from the object “2.5” do attract the particles from the object “Blender”.
- The *Strength* of the force field determines the time the particles need to reach the target. Set it accordingly.

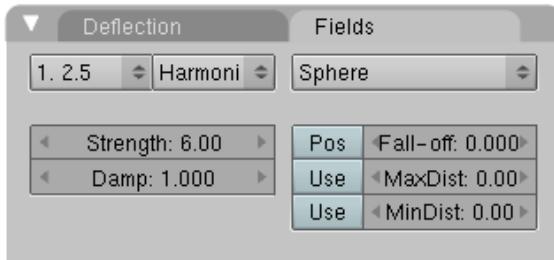


Image 3b: The particle system gets the force field.

- The *Damp(ing)* makes sure, that the particles move to their target but not beyond.

Do the same for the object “The big leap ...”, i.e. select the particle system and give it a *Harmonic* force field.

I have used a *Wind* field in the last frames to blow the particles away.

(Noobie Warning: When i followed the tutorials up to this point, i didn't know how to use wind. I thought selecting *Wind* instead of *Harmonic* would create the effect but then blender froze on me (I am guessing it went away and calculated wind for every single particle). so don't try wind yet. Wind is instructed in later tutorials.)

Links

- The Blend file to download
- The manual about force fields

3.27 Billboard Animation

Billboard visualization of particles - and especially their animation - is one of the more arcane concepts in Blender. We're going to shed some light on this. Billboard visualization is extremely powerful, everything that can be done with a halo can also be done with a billboard. But billboards are real objects, they are seen by raytracing, they appear behind transparent objects, they may have an arbitrary form and receive light and shadows. They are a bit more difficult to set up and take more render time and resources.

Billboards are aligned square planes. If you move an aligned billboard in a circle around an object, the billboards always faces the center of the object. The size of the billboard is set with the *Size* of the particle.

Texturing billboards is done by using uv coordinates that are generated automatically for them.

- The main thing to understand is that if the object doesn't have any UV Layers, you need to create at

least one in the objects Editing buttons for any of these to work (**Img. 1a**).

- Moreover, material should be set to UV coordinates in the Map Input panel (**Img. 1b**).

3.27.1 Splitting a texture

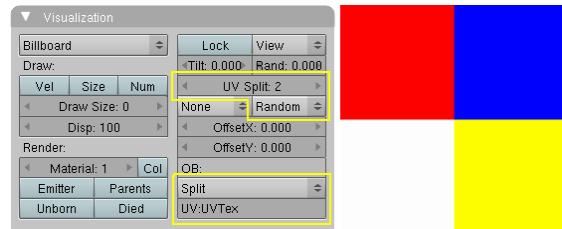


Image 2a: Billboard settings and texture for UV split

The simplest thing we can do is to give every billboard its own fraction of an image texture. We're going to use a simple image (**Img. 2a**, right hand side) and split this in four parts (*UV Split* = 2, horizontally and vertically).

- Create the UV coordinates (simply click on *New* next to *UV Texture* in the *Mesh* panel).
- Create a particle system.
- Activate *Billboard* visualization.
- Set *UV Split* to 2.
- *Offset: Random* to randomly choose one of the images sections. *None* would give only the first of the four fields, so all billboards would be red, *Linear* would first use red, then blue, then white and then yellow repeatedly.
- Set *UV Channel* to *Split*. This creates the necessary UV coordinates, that are stored in the UV layer. If you use another UV channel, you need a second (or third) UV layer to store the coordinates (see below for an example).
- Fill in the field *UV* with the name of the coordinate set (here “*UVTex*”).
- Now create a material for the particle emitter.
- Add a texture if not there already.
- Set *Map Input* to *UV*. If you only have one active UV layer this is used by default, but if you want to continue this tutorial you have to fill in the name of the UV set also (i.e. “*UVTex*”).
- Load the texture as Image texture.

Now every billboard gets a random section of the texture (see **Img. 2b**).

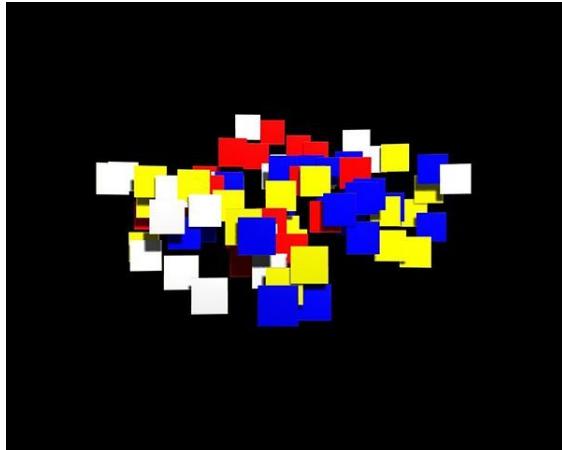


Image 2b: UV Split for Billboards

3.27.2 Animating Alpha in relative particle time

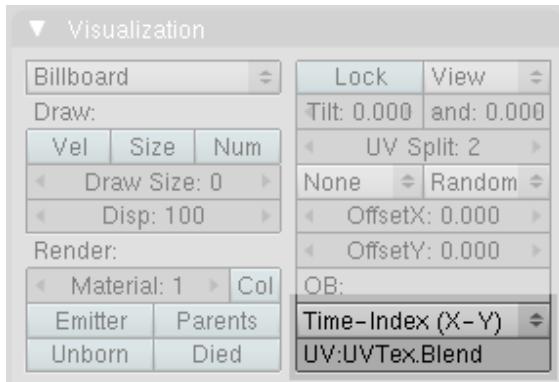


Image 3a: Using an UV layer for time animation

Now we're going to combine the texture split effect with an animation of the billboard alpha. This is done in relative particle time, i.e. in the lifetime of each particle.

Billboard time is setup completely different than particle time.

- Create a new UV layer (the second one). Name it “UVTex.Blend”. Naming is not important here as long as it is unique.
- Select the UV channel *Time-Index (X-Y)* in the *Visualization* panel of the particle system. Fill in the name of the newly created UV layer “UV-*Tex.Blend*”. This creates an additional set of UV coordinates in this UV layer.

Now both UV layers with different UV coordinates will be used. The material settings:

- Activate *RayTransp* and *TrAShadow*.

- Add a second texture to your material.
- *Map Input* also *UV*, but now use the UV layer “UV-*Tex.Blend*”.
- *Map To:*
 - *Invert Alpha* and *Invert Spec*
 - *No RGB* (since we're going to use a colorband texture)
 - *DVar = 0* to set Alpha to zero where the colorband is black.

This is one of the possible combinations for using a texture to make a material transparent (**Img. 3b**), there are several other methods that work as well.

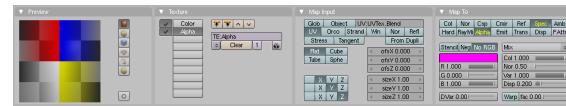


Image 3b: Material for the Alpha animation. This is already the finished preview with the texture.

The texture is really the most important thing here.

- Use a linear *Blend* texture.
- Activate *Colorband* for full control.
- Adjust the colorband to your liking.

I've mapped the texture so, that “white” means full opacity, black full transparency.

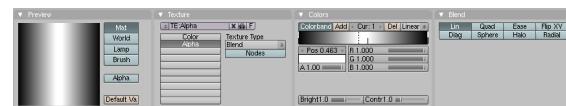


Image 3c: Texture to control the animation time.

See this animation for the result of the combined textures.

3.27.3 Animating billboard color

I will show here a second example for animating with a billboard texture, because this is so important. We're going to change the color during the lifetime of the billboard, but you can animate every property that way that can be influenced by a texture. Additionally you can animate the material itself, this is done in absolute time for billboards. So you can mix relative and absolute time animations.

- Open a new file.
- Remove the cube and add a plane.

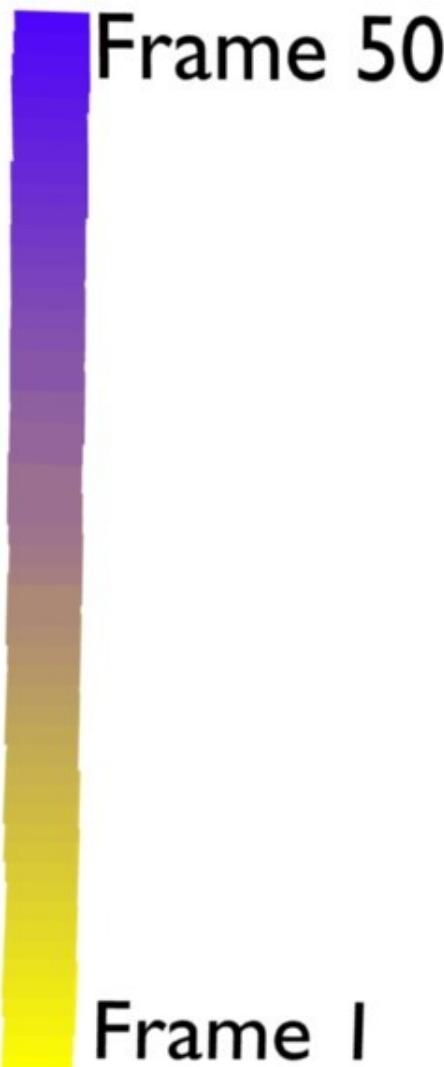


Image 4a: Animating Billboard color. Animation

- Add an UV layer to the plane
- Add a particle system with *Billboard* visualization and *UV Channel Time-Index*. Fill in the name of the UV layer.

The particle life time is 50 frames.

- Add a material to your plane.
 - Basis color *Yellow*.
- Use a linear blend texture. If you would use a color band here, you could create an even more colorful animation.
 - *Map Input UV*
 - Fill in the name of the UV layer.
 - *Map To Col*. I've used a dark blue color here.

Now the linear blend texture sets the amount of blue color that is mixed to the basis color of the material.

- At the left hand side the blend texture has a value of 0, so zero blue is mixed to the base color in the first frame of the particle lifetime.
- At the right hand side the blend texture has a value of 1, so 100% blue is mixed to the base color in the last frame of the particle lifetime.

3.27.4 Changing the starting color of a billboard



Image 4b: Flipped XY for the blend texture

If you flip the Blend texture, you get a different effect. Now every Billboard gets another starting color, but keeps that color during its lifetime.

Have I mentioned that you may use multiple textures on a billboard, combining all effects?

3.28 Soft Body Animation

Due to rapid updates in Blender, some menu items may be different from those mentioned here. If the menus don't agree then just do what seems logical.

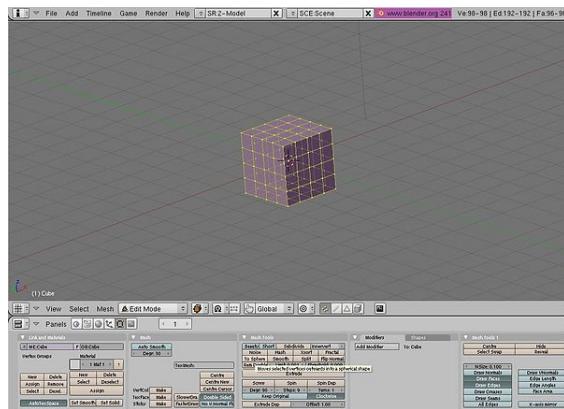
Softbody makes each individual vertex its own object that falls according to gravity and reacts to outside forces like fields. By adjusting the settings, you change the behaviour of the edges that connect the vertices. For example, you can make it so edges can stretch really far (aka elasticity), allowing the vertices to become distant, or you can make the edge stiff, so the vertices will always stay the same distance apart.

To put this in perspective, picture two cloths, one elastic and one cotton. The elastic one has edges that can extend, so if you view them in wire-frame (with vertices and edges visible) you would see the edges are more extended than an equal distance. The cotton one would only

stretch a little bit, so the vertices would stay essentially the same distance apart.

We are going to make a big rubber ball, but not a big bouncy one, a flat (and somewhat lifeless) one. Start with a sphere. I would use a cube sphere or an icosphere, UV spheres don't deform well as they have too few vertices. A cube sphere is made by subdividing a cube and doing a "To Sphere" in the Edit window, under Mesh Tools.

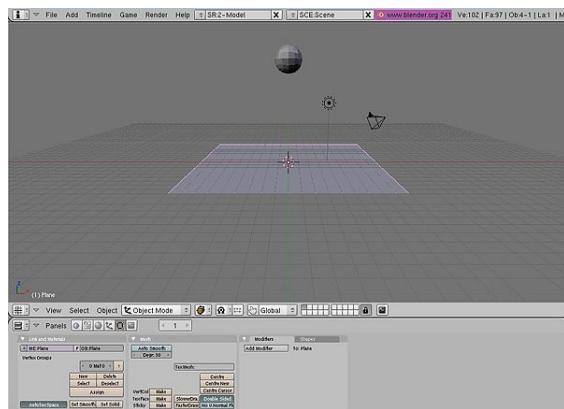
Move the sphere up and place a plane below it. Make sure to do this in the right views so that it is aligned properly. Gravity acts on the z-axis (sphere should be above the plane relative to the z axis).



Now for the soft body select the ball and go to the object tab then the physics subtab or whatever your version has. Click "enable softbody" and then turn up 'Grav' to 9.8. Click off use goal. Press the > arrow key (a few times) and you should see the ball fall. The center will remain in place but this is not a problem. If you are on a slow machine you will notice lag. This is because blender moves it vert by vert, not efficient.

Note: in Blender 2.49, you must deselect the "Use Goal" button to release the center of the ball. Otherwise, it will just hang there.

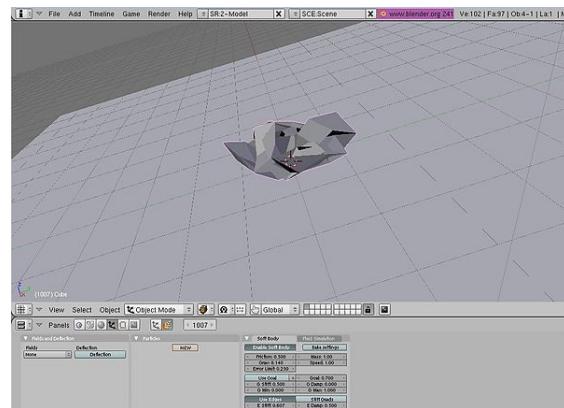
Note: In Blender 2.44, click F7 on keyboard, select the "Physics Buttons" button, select "Soft Body".



When it reaches the plane it will pass through. To fix this

we must make the plane affect the softbody. To do this make the plane deflect in its physics buttons.

Noob Note: *To do this in version 2.49 select the plane, go to object buttons -> physics context -> collision subcontext and select Collision. The variables you can play with here are under the Soft Body and Cloth Interaction.*

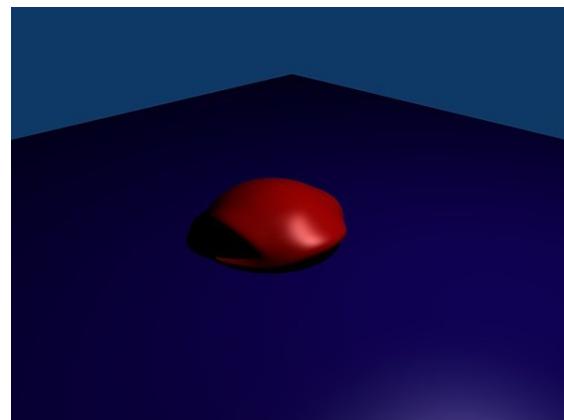


Now the ball collapses into a strange quivering wreck after impact. To fix this, you need to turn on the stiff quad button, but set the edge stiffness down a bit, so its more bouncy. You can use the bake function to solidify the settings see below

Noob Note: *You might have to turn up the Rigidity Level to 0.100 (in the Soft Body Tab) as well in order to prevent the object from collapsing. (I used a subdivided cube as Object)*

What I recommend to do before rendering as animation: In the Bake Settings (Soft Body Tab) Set Interval to 2 or 1, so the object will not start deforming too early before impact. This will slow down the bake process - just slightly - but make the object bounce more dynamically. Then bake again.

Noob note: In blender 2.46 you have to adjust the value of Be in the soft body tab, I changed it to 0.4



another thing you could do is animate the scene. Just put up animation video and set frames at about 50. Go to object mode, select sphere and press i key, and select

LocRocScale. =)

3.28.1 Explanation of Settings

I invite you to correct and expand these definitions:

Softbody

- **Friction:** creates a resistance to movement of the whole object, like being submerged in a viscous fluid
- **Grav:** the rate of velocity change due to gravity. Results in a constant -z force.
- **Mass:** (Force = mass × acceleration) affects everything by making the object heavier.
- **Speed:** tweaks the simulation to *run* faster or slower.
- **ErrorLimit:** raise it and the simulation will solve faster but strange things might happen. Save frequently, as blender might go nuts with this or any physic simulation (but less so after 2.4)
- **Goal:** makes the object try to return to its original position, useful at times, in the tutorial you could turn gravity off and key the ball falling and use this to keep it a ball.
- **use Edges:** uses the edges a means of resistance to movement for the object. Helps to keep it looking possible

3.29 Simple Cloth Animation

In this tutorial, we will be making a simple skirt, and using the cloth physics system to make it fall in realistic-looking folds.

3.29.1 Making the Skirt Mesh

1. Open Blender and delete the default cube, if you aren't looking down on the scene, press NUM7 .
2. Shift + A → Add → Mesh → Circle
3. The circle will be created, and in the Tool Shelf on the left you should see a panel appear for adjusting its settings. Set the number of vertices to about 12.
4. Switch to Edit Mode with Tab . All the circle's vertices should be initially selected; if not, use A to select them all.
5. Press E to extrude a second copy of the vertices; press S Shift + Z to scale the extruded vertices in the X and Y directions. These will make up the hem of the skirt; scale it out to as large as you like.

Note that this is positioning the skirt out flat horizontally, instead of hanging down as you would expect; Blender's cloth animation system will take care of that, and this positioning gives maximum opportunity for the skirt to fall in dramatic folds.

6. Now we will need to subdivide the mesh. The physics can only act on actual vertices, so the more of these we have, the more realistic the cloth effect will be. Select all vertices in the skirt, press W and select the "Subdivide" option. A panel will appear in the Tool Shelf for controlling the settings for the subdivision operation; set the number of cuts to, say, 4.

7. While you're at it, switch to OBJECT mode and look in the Tool Shelf for a pair of "Shading" buttons titled "Smooth" and "Flat"



, and click on "Smooth".

3.29.2 Creating the Vertex Group

1. Now we have to specify that the waist of the skirt will stay fixed in place as it falls: deselect all vertices, and select the innermost ring of vertices. The quickest way to do this is to hold down Alt + Shift and click with RMB on one of the edges bordering the hole in the middle. With the entire ring of vertices selected, go to the Mesh Object Data context in the Properties window, find the "Vertex Groups" panel, and click the + sign to create a new group. This will initially be called "Group"; perhaps give it a more meaningful name (like "Waist"), and click "Assign" to put the selected vertices into the new group.

3.29.3 Animating the Skirt

1. Tab out of Edit Mode into Object Mode. The skirt should still be selected.

2. Go to the Physics tab in the Properties window (last icon in the row at the top). This will just show a few buttons to begin with



3. Click the "Cloth" button; a whole lot of other settings should appear, most of which can be left at

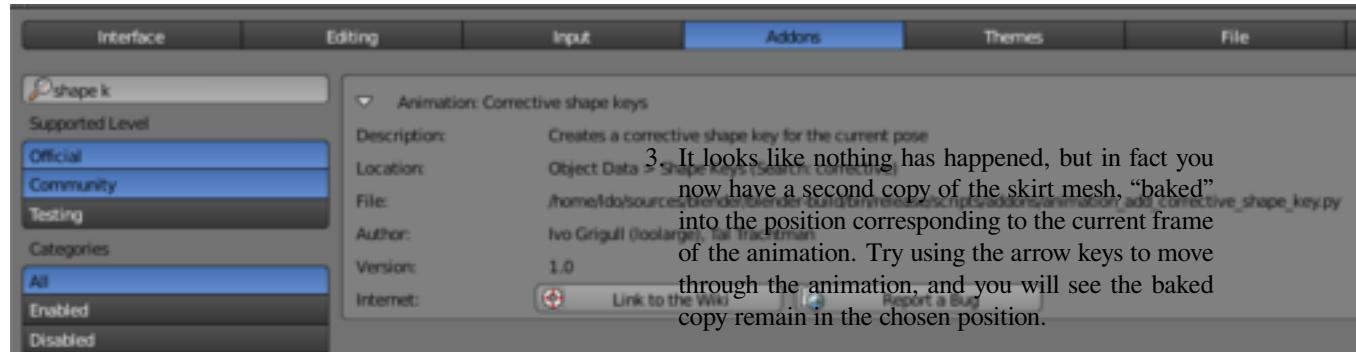
their default values (make sure the checkbox at the top of the “Cloth Collision” panel is checked). However, check the box titled “Pinning”; now you can click in the field just below it to bring up a popup menu of all the vertex groups in the mesh; this should just contain the one entry named “Waist” you created earlier, so select that.

4. Now the magic happens ... rotate the view to an oblique one to give yourself a good view of the process, and hit Alt + A . You should now see the skirt fall from its horizontal position to a more natural vertical one, developing some folds in the process.
5. After the animation has run through at least one complete cycle, hit ESC to stop it.

3.29.4 Prior to Keeping the Folds

Before doing the next step, we need to enable one of the standard addons that come with Blender. This will let us make a copy of any stage of the physics simulation into a separate object.

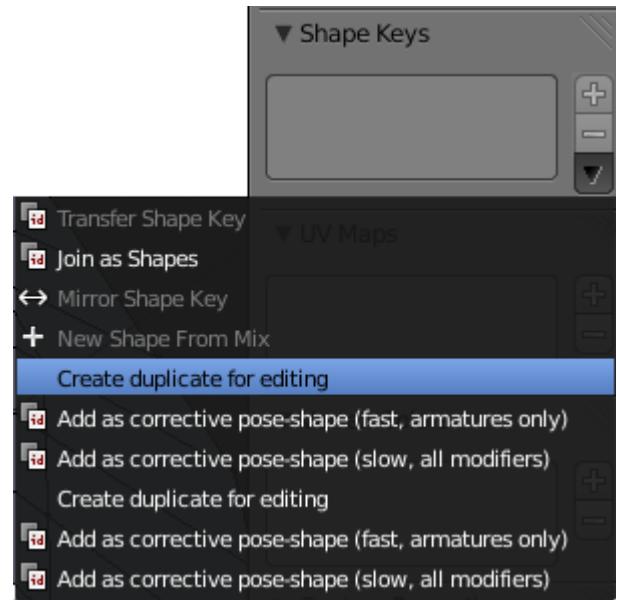
Go into the User Preferences window and bring up the “Addons” tab. Look for the “Animation Corrective shape keys” addon (typing “shape k” into the search box should be enough to find it).



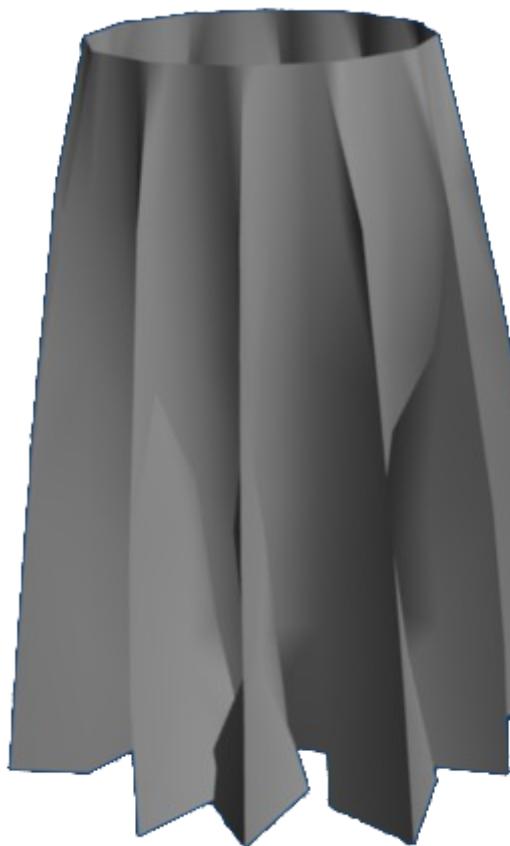
Enable it. Now back to the 3D view...

3.29.5 Keeping the Folds

1. Use the left- and right- arrow keys to step through the animation one frame at a time, until you find a position for the skirt that you like.
2. When the 3D view is showing a nice shape for the skirt, go to the mesh data tab in the Properties window, and look for the “Shape Keys” panel. Click with LMB on the down-arrow just below the + and - signs, and in the menu that appears, you should see the item “Create duplicate for editing”
3. It looks like nothing has happened, but in fact you now have a second copy of the skirt mesh, “baked” into the position corresponding to the current frame of the animation. Try using the arrow keys to move through the animation, and you will see the baked copy remain in the chosen position.
4. At this point, you can delete the original animated skirt mesh (or move it to another layer for future reuse), leaving the nicely-folded copy.



Select that.



3.29.6 Extra Practice

1. You will notice at some points during the animation, the folds of cloth pass right *through* each other, which is of course impossible with real cloth. To prevent this, you could go to the “Cloth Collision” panel in the Physics tab, and click the “Self Collision” checkbox. Rerun the animation (Alt + A) to see the difference; what other effects does it have?
2. Maybe the folds don’t look realistic enough. Go back to the original mesh, bring up “Subdivide” again and subdivide it by a couple more levels. Rerun the animation (Alt + A). It should take a bit longer for the first cycle, but do the results look better?
3. This YouTube tutorial might also help: <http://au.youtube.com/watch?v=mgYhZ3hWwTQ> happy animating!

3.30 Soft Body with Wind

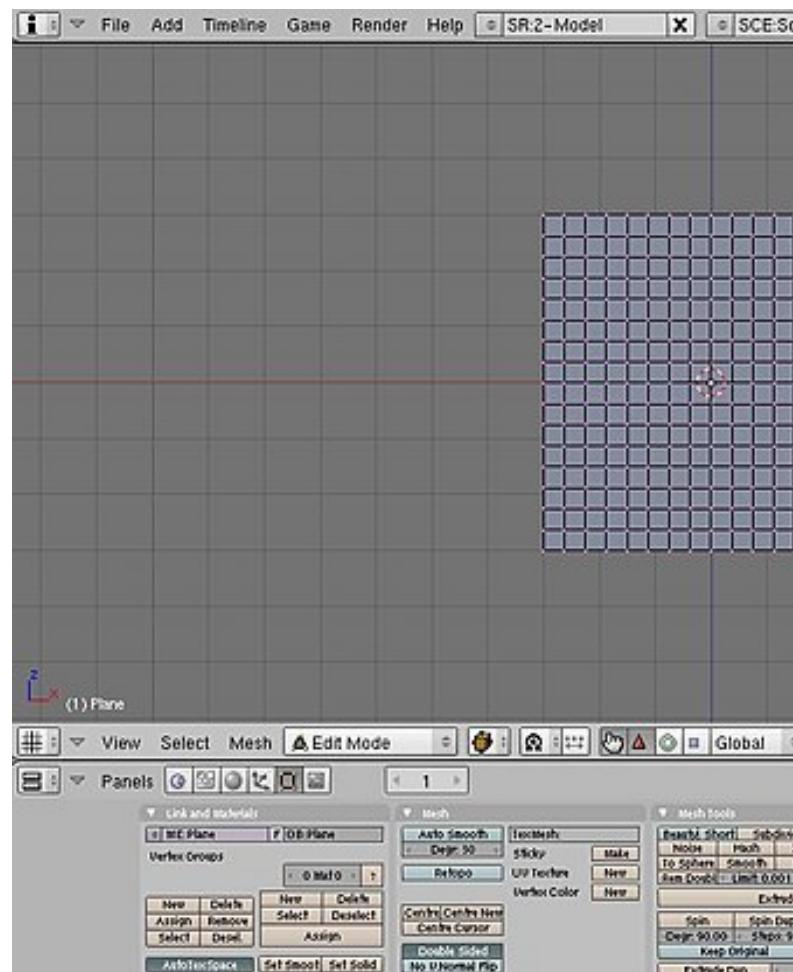
3.30.1 Prologue

This is the **Blender wind and soft body tutorial**. This tutorial will try and help your knowledge of using Soft Body and the Fields and Deflection panels in Blender 3D. (For best results, I recommend you use Blender version 2.43 or higher). Don’t forget to save your work at various points throughout the tutorial.

3.30.2 Setting up scene

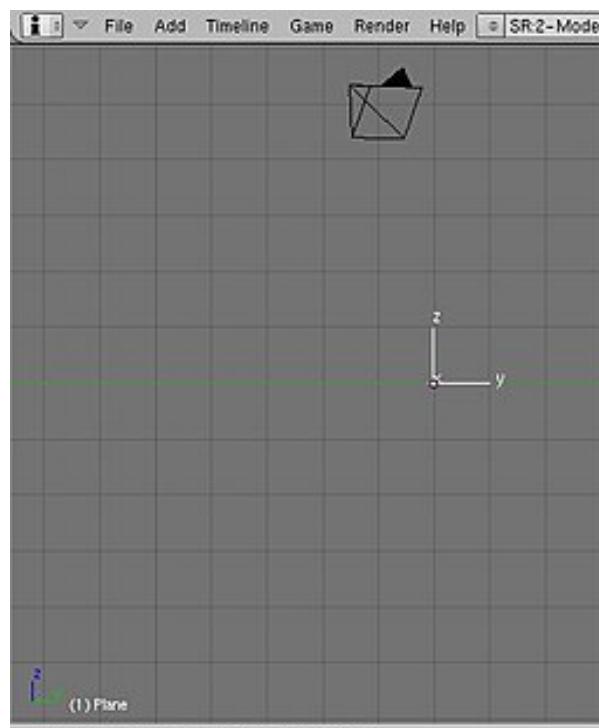
The plane

Delete the default cube [X key, or yes, even the Delete key] and enter Front view mode [Num. Pad 1]. Add a plane, scale 3 times [S key, 3, Enter] and sub-divide 4 times until you get something like the shape below. If your computer can handle more, and you want more, subdivide as many times as you like, but if your PC is struggling with this, undo once or twice.

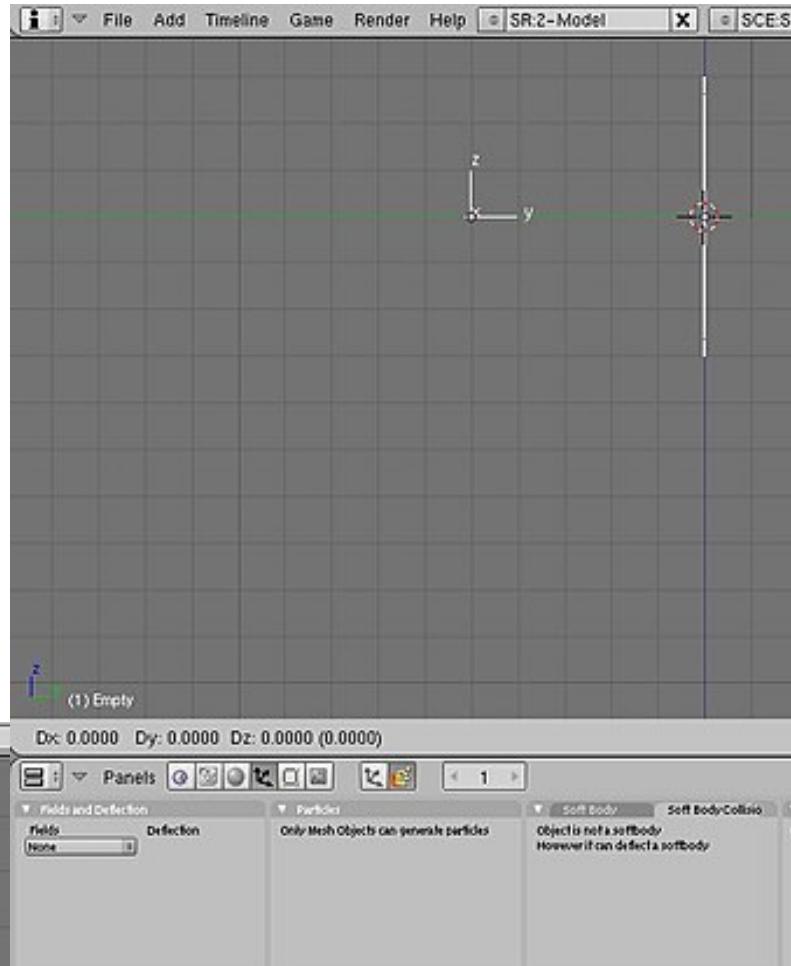


The wind items

TAB out of edit mode (if you haven't done so already), enter side view [Num Pad 3] and add an empty. Clear the rotation [ALT + R] and rotate it 90° [R, 90]. Place it about -5 from the centre $[G, y, -5]$ and make sure its still in the middle of the X axis. Your scene should look like the picture below:



For the second wind item, add an empty like before and clear its rotation, but DON'T rotate it any more. Move it along the Y axis by 4 places $[G, Y, 4]$ and -10 by the Z $[G, Z, -10]$. And now your scene should look something like this (with all items selected).



And that's all your items you'll need for this. Onto the harder part.

3.30.3 Designating each object's job

The plane

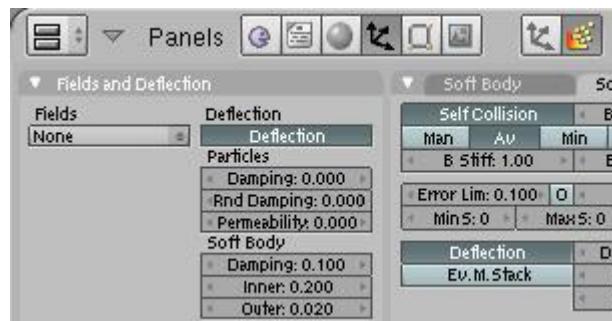
Now, the plane will become a soft body, so enter the Object Panel [F7], click on the Physics Button (beside the three arrows, see second picture below) and click on the soft body button. You will get a load of boxes, but we only need to look at the Grav, Goal, and the bottom parts:



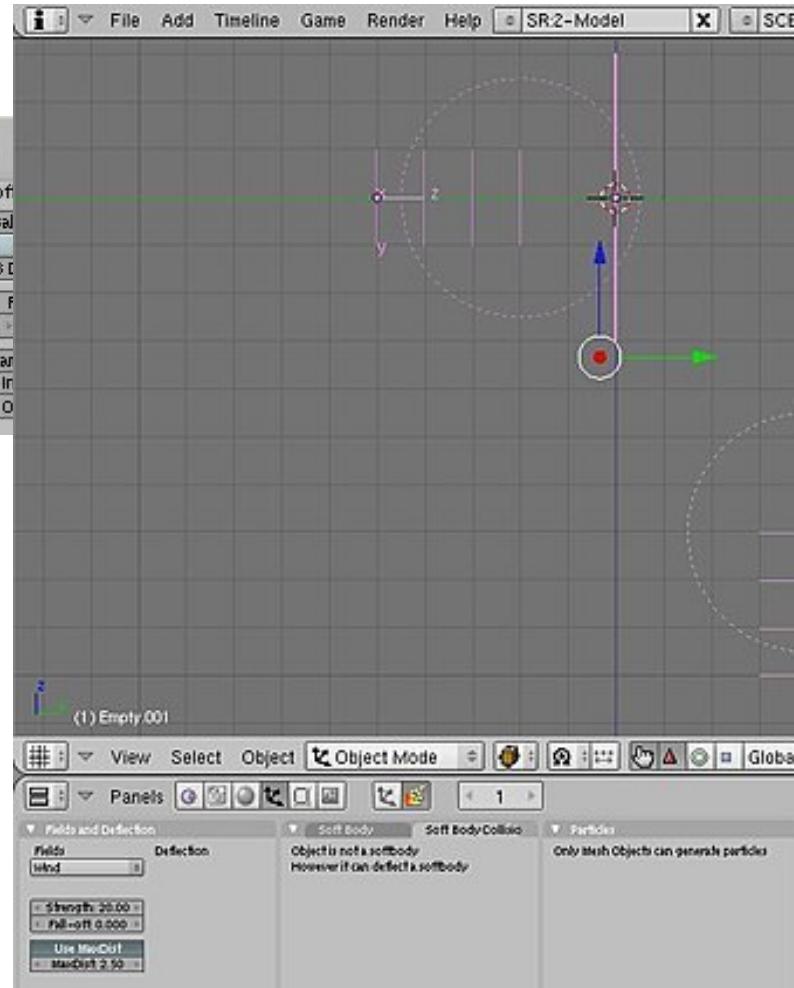
Change the values of these as shown above. Change the Aero part to 732 (if you go up to 1000, it changes the final output by a bit).

Now, click on the Soft Body Collision tab and click on

the Self Collision and Deflection buttons. You don't need to change any of the numbers here. The buttons window should look something like this:

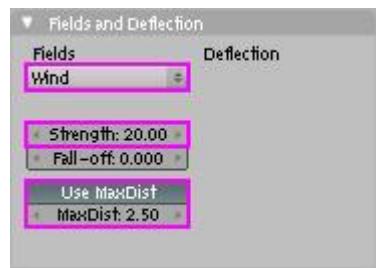


Now we will move on to the Empty items which will be the wind.



Creating the wind

The two empty's we created earlier will be the items that will act like wind, and when you render at the end, they will be invisible, just like the wind. Anyway, select one of the empty's and enter the Physics panel. Click on the box under the Fields and select "Wind". (Blender 2.5+ note: First you will have to choose "Force Field", then change the Type to "Wind".) Change the Strength to 20, click on Use MaxDist and change MaxDist to 2.5.



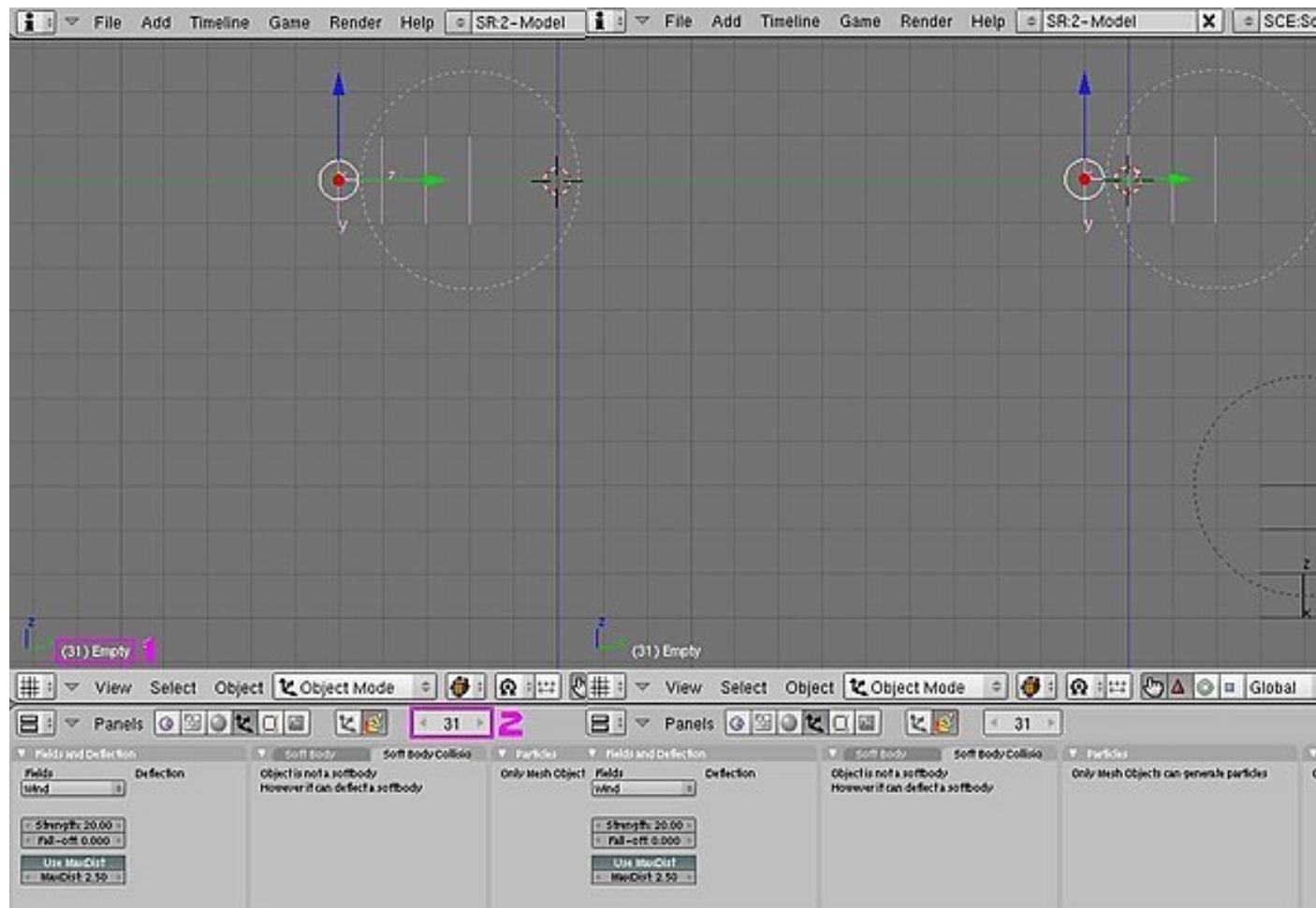
Do the same for the other empty. Your scene should look something like this:

3.30.4 The movement

This is probably the most complicated part, but shouldn't take long.

The Empty on the left

In side view [Num pad 3], select the empty on the left and do the following: 1. Press [I key] and then select LocRotScale from the list it gives you. This inserts a key frame, saying that on this frame, keep the Location, Rotation and scale the same throughout the rest of the movie, unless it comes across another key frame, which might tell it to move or stay the same. Now, press the Up Arrow Key until you get to frame 31 (about 3 times). If you don't know where the box is that tells you what frame you're on, check the following picture:



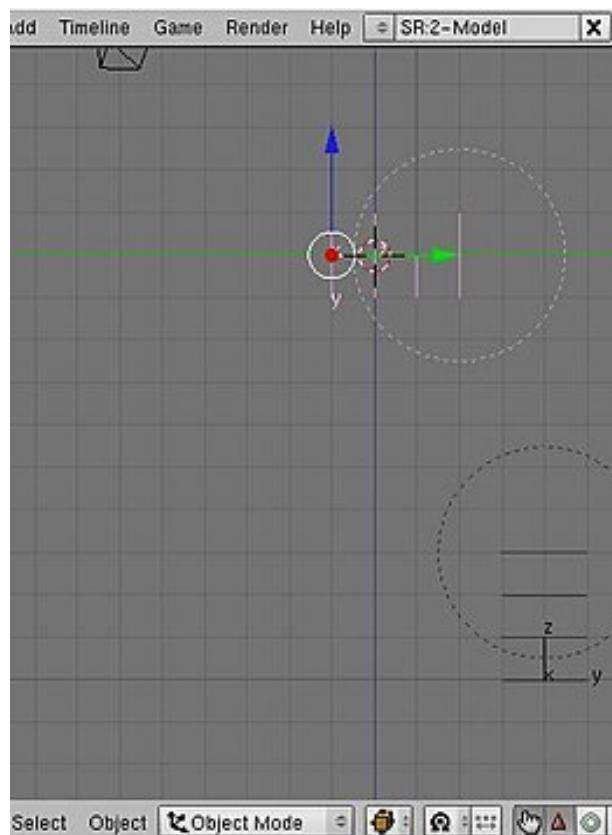
Number 1 shows the current frame number and the name of the selected object while number 2 shows the current frame and allows you to change the frame with the arrows on either side.

Now, click on the box (shown below) and select “IPO Curve Editor” from the menu.

Noob note In Blender 2.6+ the **IPO Curve Editor** has been replaced by the **Graph Editor**, which allows you to edit F-curves.

Back to the movement

On frame 31, select the empty on the left (the one selected in the picture above) and press [G key]. Now press [Y key] and press “4”, then [Enter key]. Now press [I key] and select LocRotScale again. Now, if you press and hold the Left Arrow Key, you should see the Empty move back to its starting place. Now, to check the curve guide (called the “IPO Curve Editor”); right-click on the top panel (number 1 in picture below), then select “Split Area” and click when the line is a bit out from the side (number 2 in picture below).



This window just shows where the key frames you entered are, and how much the shape changes, moves or rotates. Now, go back to frame 1 and press [ALT]+[A], which plays the animation. You should see the “wind” hit the sheet and the sheet will blow away. Pretty nice, huh? Now, it would be fine like that (I’ll show you how to get rid of the blockyness in the last part), but say you want to hit the sheet again and send it upwards? We will use the second empty for this.

Select the second empty (the one on the bottom). Insert a key frame ([I key], LocRotScale) on frame 1 and frame 50. Go to frame 60, grab the empty and move it up 8 places ([G key], [Z key], 8). Insert a key frame ([I key], LocRotScale) and then go back to frame 1. You have completed the basics, and the most of this tutorial. Now, press [ALT]+[A key] and watch your animation.

3.30.5 Finishing touches

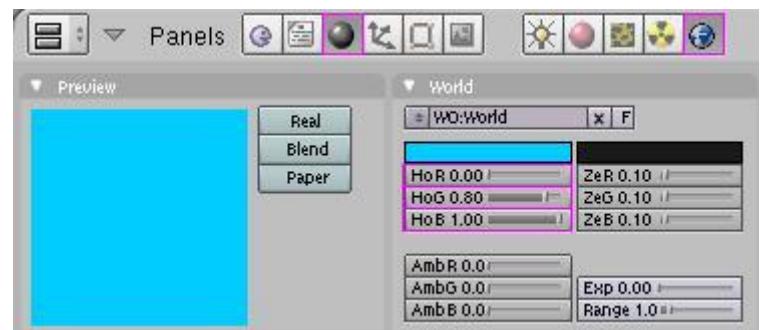
Blockiness

Now, to sort out the blockiness, go to the Editing buttons panel [F9] and select the “Set Smooth” button.

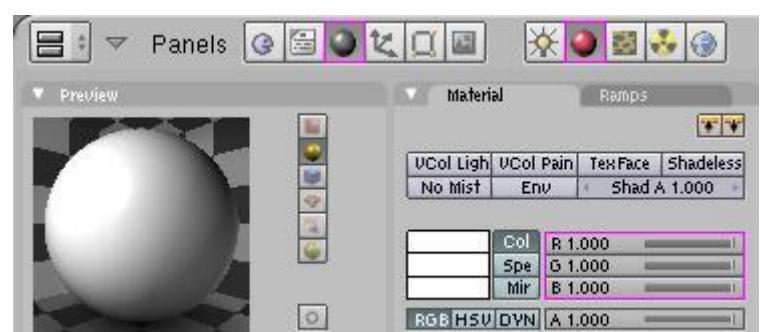
Now if you preview again ([ALT+A]), you should see the plane is smoother, but might be a bit unrealistic (i.e. not acting like real cloth). Now that can’t be helped, the only way to get rid of it is to **Subdivide** the plane several more times, but that will put a **LOT** of strain on your computer, so I wouldn’t recommend doing it unless your computer can handle it.

Background

First of all, got to the Scene panel [F10] and over at the Format panel, change SizeX to 400 and SizeY to 300. Now, to change the background colour when you render, go to the shading panel [F5] and select the World buttons. Now, in the World panel change HoR to 0, HoG to 0.80 and HoB to 1.0.



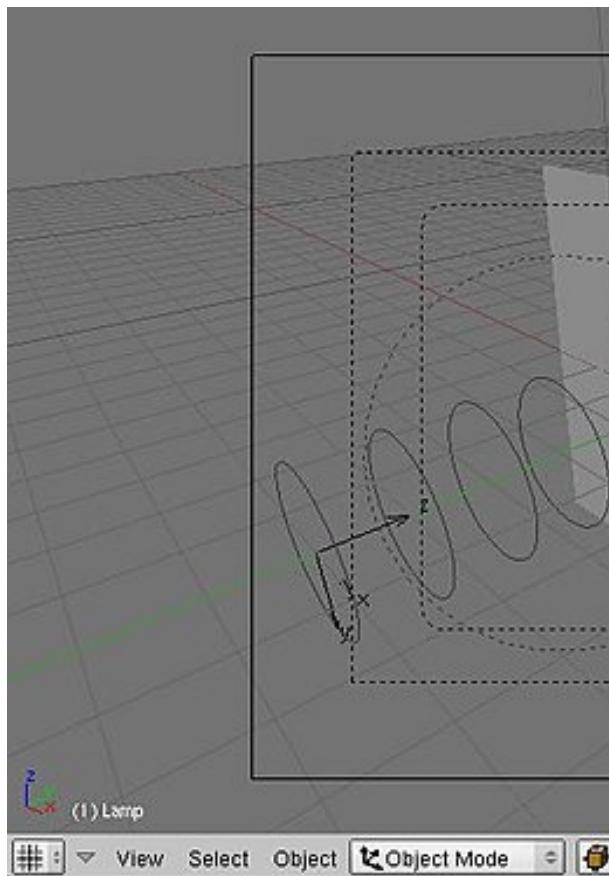
All that does is just changes the background colour to a brighter, nicer shade of blue. Now, I’m just getting a bit lazy, so I’ll just explain this part with a picture: (don’t forget to select the plane).



Change the values and press the buttons shown in the picture. This will change the colour of the plane to white.

The only problem now is the light. Press [F12] and you should see what I mean. There’s no light shining on the

plane where the camera is, so its just shaded dark. To fix that, select the light and go to the shading panel [F5]. You now have options for the light. Click on the “Sun” button and press [ALT]+[R] to clear rotation. Now, in side view [Num pad 3] press [R] and then type -45 . Enter top view [Num pad 7] and press [R] and then -45 . This now shines the light in the direction of the plane, but, if like mine, the light is above the plane in top-view, press [G], [Y] and then -5 . The line coming out of the light source should be roughly pointing at the centre of the plane. Here’s what your window should look like:

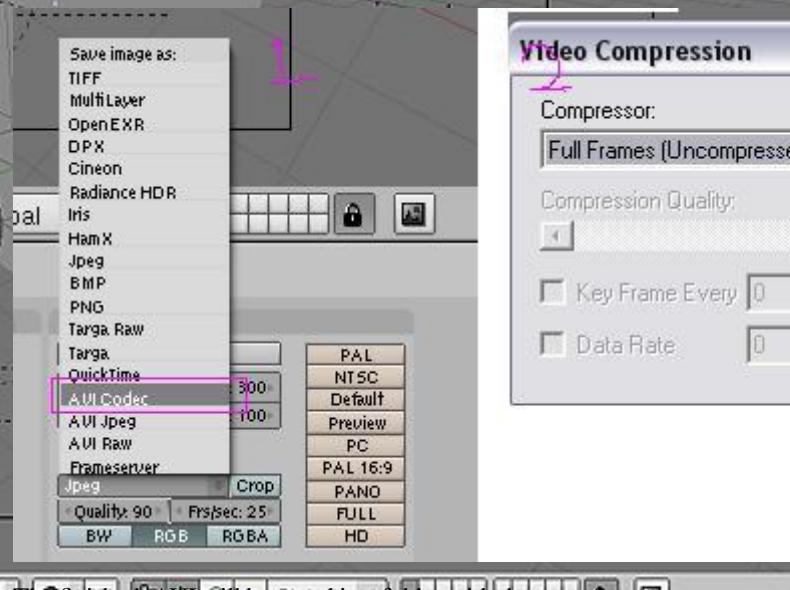


The pink line is the light. This was taken in camera view [Num pad 0]. Render the frame and you should see what the final product looks like. Move to any frame you want and press [F12] and it will show the rendered image for that frame. Now, rendering the entire animation might, in fact, will take a long time unless you have a completely new PC, are using a big server like DreamWorks (one of the big PC's, used for rendering images faster) or have a quicker renderer, I don't use any external renderers, and am stuck using a PC that is about 2 years old (in other words, full of junk). It took my PC less than 5 minutes to render 100 frames, and the last 25 were just blank (as the plane just flew above the camera). To change the amount of frames to render, and to render the full animation, here's what you press:



The box that says End:100 lets you select what frame you finish rendering the animation on (NOTE: the Sta:1 button lets you select what frame to start rendering the animation). When you press the ANIM button, it starts rendering the animation. The Box covered in blue, that says Jpeg, lets you change the format of the image in the end.

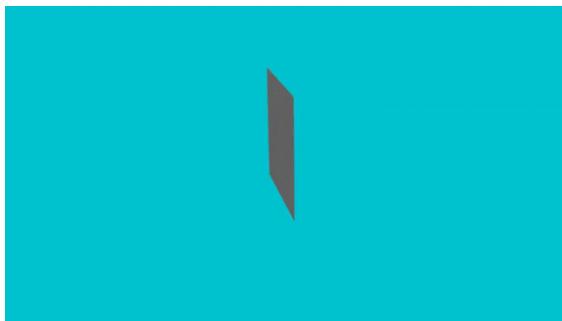
To export the animation as an AVI format, select the AVI button shown below, then press OK. Now, you need to Animate the sequence again; so press the ANIM button again.



The finished AVI will be saved in a folder with the rest of your rendered images. It should be saved in a folder called tmp, which (on a Microsoft computer) is in My Computer/tmp. The name of the file will be whatever your start and end frames are, so in this movie, mine is called 0001_0100.avi. You can now do whatever you want with your movie now, and hopefully you have learned a trick or two that will be helpful later on in your Blender career.

3.30.6 Final result

Here is an animated .gif of my final result: (imported into Macromedia Flash and exported as an animated .gif) You are required to be an auto confirmed user or uploader. I made a gif with a frame-step of 4.



3.31 Blender Game Engine Basics- Rolling Ball

3.31.1 Introduction

The Blender Game Engine is an interesting feature of Blender. It is basically a 3D environment in which 3D objects move around and react to each other upon contact. One common application is to recreate 3D architectural tours.

In this tutorial, you will learn the basics of object collision within the Blender Game Engine (BGE). From Blender games to use in animations, the bullet physics engine offers a massive number of possibilities. The tutorials found within this wikibook on the subject of the BGE are generally focused on game creation, but the concepts taught within can be applied to a multitude of situations.

As a start, we will teach you to make a ball roll realistically down the hill using Blender's game engine.

3.31.2 Adding the Hill

First, make a plane, then switch to Edit mode (**TAB**), and multi-subdivide it with 2 cuts (**WKEY** → *Subdivide Multi* → 2). Next, enter face select mode (**CTRL+TAB** → *Faces*) to drag the center face up, in order to form a rough outline of a hill. Add a subsurf modifier (in the edit buttons) to about 3, then apply. You should now have a serviceable, but small, hill. Scale the hill up (**SKEY**) by about 10 times, and we're ready to add the ball.

Noob note:

1. You can also use a 3*3 grid.
2. Delete the original cube first.
3. The subsurf modifier is not essential.

3.31.3 Creating the Ball

Now, add an icosphere (**SPACE** → *Add* → *Icosphere*) and relocate it to be just above the hilltop (**GKEY** or use the translate widget by pressing **CTR+ALT+G**). Let's

change the color of the sphere so we can differentiate it from the plane. Go to the material buttons (with the sphere selected) and click on the white panel beside the **COL** value. In the color selection wigeet that appears, change its color to a bright red.

Next we need to make the Physics engine iterate over it. With the sphere selected, go to the logic buttons (the little purple Pacman-icon). **NOTE: In Blender 2.5 and above the Logic Buttons are gone. In order to have the 'Actor' button, click on the button showing 'Blender Render', and select 'Blender Game' Engine. Then go to the physics tab in the buttons menu. There you will see the 'Actor' Button.** You will see a button in the top left that says *Actor*. Press it. Now select from the selection box beside of the "Actor" button *Rigid Body*. This makes the ball roll, instead of staying completely upright the entire time. You will see a bunch of settings available now. Change *Radius* to 2. This changes size the physics engine *thinks* the ball is. You notice a dotted circle around the object; this is the boundary. Now change the *Radius* back to 1. You now have your first Blender game ready to go.

Noob Note:

1. Make sure you are in object mode first before you add any object.
2. **F4KEY** is the shortcut to the logic panel.

3.31.4 Testing your game

Now the time has come for the first test of our game.

1. Add a light source well above the hill (**SHIFT+AKEY** → *Lamp* → *Lamp*). Align in front view (**NUM1**)
2. Press **NUM5** to switch to Perspective mode, which gives a realistic view, rather than a view in which objects stay the same size with distance (be sure to switch back to Orthographic view when you are editing again using **NUM5**)
3. Enter textured mode (**ALT+ZKEY** -- press **ZKEY** to switch back to solid view mode)
4. Switch into side view (**NUM3**) and press **NUM8** several times to get a good perspective on the ball.
5. Press **PKEY** to play the game (Make sure you are in Object mode (**TAB**))
6. Press **P** to start testing the game. You should see the red ball drop onto the hill.
7. Press **ESC** to quit testing the game

3.31.5 Video capturing your game

When you press the PKEY or click game, Screenshot, Blender will play it using the 3D view. Many rendering features are not shown in this 3D Window and it does not render the view in order to get good pictures with textures and lighting. You must capture your object that keep on changing(the actor) so that it can be animated.

View -> click View button , -> next step / -> alternative step RMB -> right mouse button IKEY -> press the I key on keyboard

Split window, IPO Curve Editor View, Game, Record Game Physics to IPO,

object mode, RMB(select actor), IKEY/select key frame, Loc(location of actor object only),
PKEY(play game), ESCKEY(stop game)

scene(F10), output, f:\animationball (your file name), stamp, time, date, draw stamp, format,(choose your output format)

anim, end(ending frame), 270, step, 10(for testing), ANIM/Render, Render Animation/CTRL F12

3.31.6 Conclusion

With the knowledge acquired in this tutorial, there are many things you could accomplish within the Blender Game Engine, although the majority of them would require more knowledge. So read on, and work your way through the multitudinous seas of tutorials (That is, two).

3.31.7 Extra Tutorials

Making a Basic Game: [Link](#), The State Actuator: [Link](#), Blender Game Engine Mouse Follow: [Link](#), Blender Bullet Physics: [Link](#), Domino Game: [Link](#), Rag Doll: [Link](#)

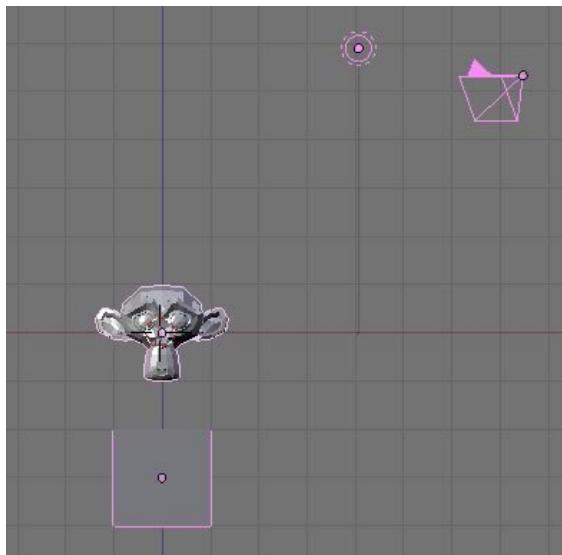
3.32 Platformer: Controls and Movement

This tutorial will teach you the basics of the Blender Game Engine, and how to create a platform game, which is a game where you overcome in-game obstacles by controlling the character's motions. (e.g. Super Mario)

3.32.1 Set Up

Actor	Ghost	Invisible	Advanced	Sensors	Set	Act	Link	State
Mass: 1.00	Radius: 1.00	No sleeping						
RotDamp 0.100								
Rot Fh	Form: 0.40	Anisotropic						

To start, switch to the frontal view NUM1 and move the starting cube to -3.0 along the Z axis, or three spaces down. If your default settings do not have you starting with a cube, create one now. (Don't forget to switch to the Blender Game engine. You can do this by going to the Properties panel and selecting the Actor tab.)



How it should look so far.

to the button on the top center of the screen that says Blender Render and switch it to Blender Game.) Next, hit **SPACE** → **Add** → **Mesh** → **Monkey** to create Suzanne the Monkey. Any mesh will do, but it's the easiest of the basic meshes to intuitively determine local direction from. Next, go to the logic window (with Suzanne still selected) and toggle on "Actor" and then "Dynamic". Hit **PKEY** to test. Suzanne should fall, then stop upon hitting the cube. Press **ESC** to exit.



3.32.2 Controls

Now we will add the controls. First, forward:

In Blender 2.59 these controls are available in the 'Logic Editor' window.

1. Create a Keyboard Sensor, AND Controller, and Motion Actuator. Connect them by dragging lines between the dots next to their names. (In 2.59 the controller is created automatically, if the other two are connected)
2. Select the empty box next to the word "Key" on the Keyboard Sensor and hit the up arrow key on your keyboard.
3. Set the dLoc on the Motion Actuator to 0.10 Z and toggle Local Transformation (may appear as simply an L next to the dLoc row).

Next, backward. Repeat the process used for forward, but set the dLoc to -0.10 Z and the key to the down arrow.

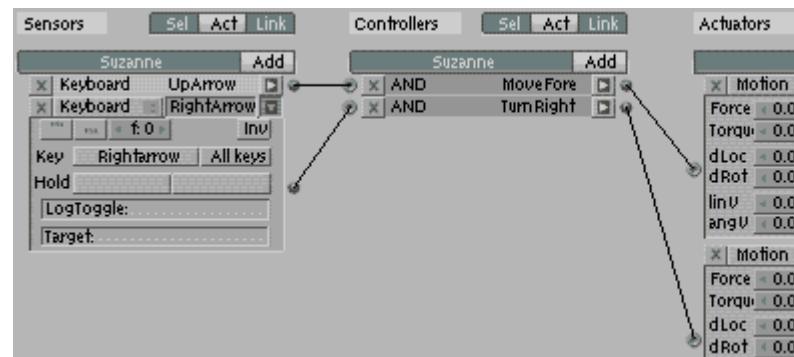
You can repeat this for left and right if you so choose to have a linear control scheme. We will not be using such a scheme for this tutorial, but you should still be able to follow along just fine. Instead, the character will rotate left and right and move in the new forward and backward.

(Note: Using an X or Y value for Dloc may be more effective as using Z value seems to just make her jump not move in any sort of direction which defeats the purpose of the final step of the tutorial)

To rotate right:

1. As before, create a Keyboard Sensor, AND Controller, and Motion Actuator, then connect them.
2. Set the key to the right arrow and set the dRot to -0.10 Y.

Repeat for left, with the left arrow and 0.10 Y.



Finally, the ubiquitous, and some say defining, Platformer move: the jump. Repeat the previous processes, with the space key as the trigger and the linV of the motion effect as 7.50 Z (not local). This doesn't actually move the character in a certain direction, but sets its velocity. If you've done programming for games before, this will probably be familiar to you. Each frame, the engine adds the velocity of the object to its current position. The gravity portion of the engine subtracts from the upward velocity each time. When it reaches a negative, it's adding a negative number to its altitude, ergo subtracting from it, ergo causing it to fall. This is why using linV to move the character does not cause it to immediately relocate to the target position.

3.33 Maze: Force and Multiple Levels

3.33.1 Introduction

This tutorial is intended as an intermediate introduction to the Blender game engine, in the form of a game, and is the sequel to [Platformer: Creation and Controls](#). It will require a familiarity with the Blender UI, simple commands (such as **AKEY** to select) and basic modeling

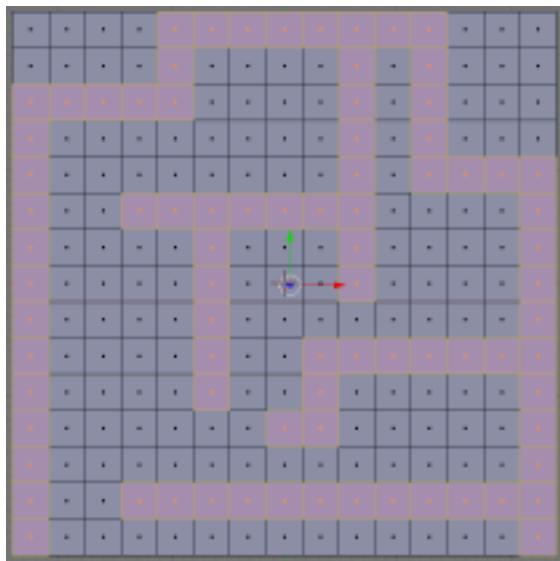
skills. The game we will create within Blender will have the following features:

1. a protagonist controlled by means of the WASD keys
2. a maze surface without walls
3. death on falling off the maze
4. multiple levels
5. dynamic obstacles
6. a goal within the maze that will transfer you to the next level

This tutorial was written for version 2.44 and has been tested for all later versions (as of April, 2008)

3.33.2 Maze Surface

Now we'll make the maze. When you are creating the path of the maze surface keep in mind that you need a start and a goal.

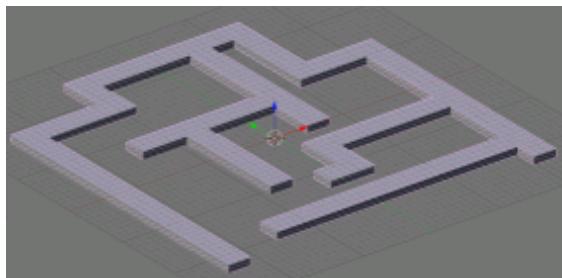


Selecting the faces

1. Clear the scene; Select all (**AKEY**), delete selected items (**XKEY** → **ENTER**)
2. Go into top view (**NUM7**)
3. Add a grid (**SPACE** → **Add** → **Mesh** → *Grid*) with X and Y resolution of 16
4. Scale grid by 24 times (**SKEY**, type 24, **ENTER**)
5. Enter edit mode with the grid selected (**TAB**)
6. Go into face select mode (**CTRL+TAB** → *Faces*)

7. Start selecting faces so as to make a two dimensional maze (use **BKEY** to draw selection boxes)
8. With the faces selected, duplicate those faces (**SHIFT+D**, **RMB**)
9. Move those faces away from the original grid (**GKEY** or the movement widget **CTR+ALT+GKEY**)
10. Delete original grid
11. Center your maze using **GKEY** or by pressing the **Center** button in the **Mesh** panel of the Editing buttons (**F9**)

You should now have a two dimensional maze. In the game engine, we only see one side of every face, so this maze will not appear to be there when seen from below, and will be obviously two dimensional when seen from the surface of the maze itself. This is undesirable so we will make our maze 3 dimensional by extruding the maze surface down.



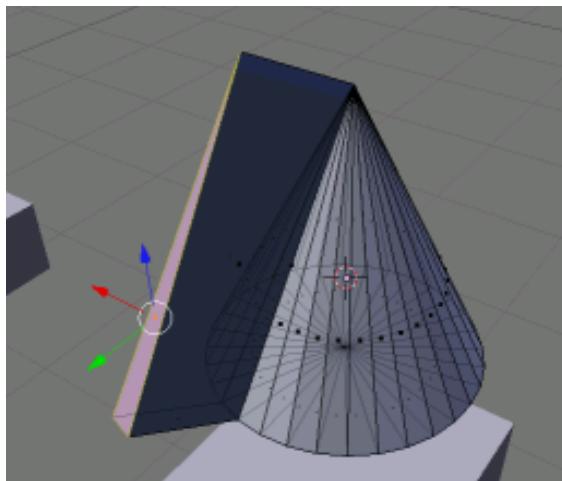
The extruded maze

1. Go into side view (**NUM3**)
2. Select all the faces (**AKEY**)
3. Extrude down, -1 unit (**EKEY** → *Region*), hold **CTR** to constrain to 1 unit increments, or manually type in the value -1, like you scaled the grid in the beginning.
4. **LMB** to accept the operation, or **RMB** to cancel the move of the faces. If you cancel it, the faces will have extruded, but will simply not have moved. You can undo the operation (**CTRL+Z**), or delete the vertices/faces you have extruded (**WKEY** → *Remove Doubles*).

Your maze is now complete, only lacking someone trapped in it...

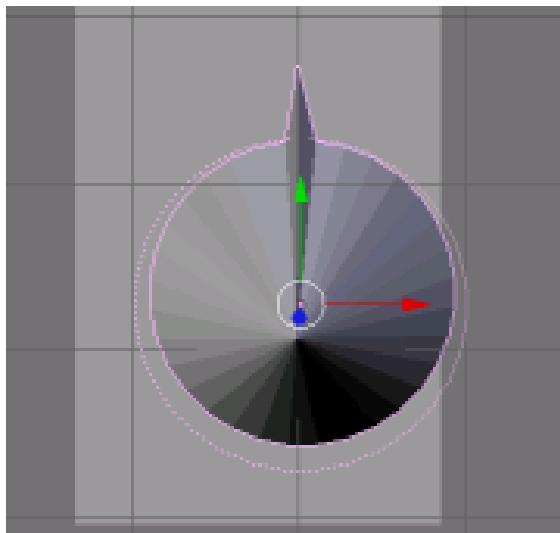
3.33.3 Character

We will create a very basic character sufficient to illustrate the concepts meant to be conveyed in this tutorial.



Extruding a nose

You can make a character as complex or basic as you desire, but it is important to always have an indicator of direction on your protagonist, otherwise the player may become confused. So, we will use a primitive with a ‘nose’ (i.e. a face or vertex extruded away from the object center in order to provide a point of rotational reference). I’ll use a cone for this tutorial, simply because they don’t roll very easily.



Rotate to the direction character will move

1. Enter top view (**NUM7**)
2. Add the cone (**SPACE** → *Mesh* → *Cone*, use default values)
3. Select cone and enter Edit mode (**TAB**)
4. Enter face select mode (**CTRL+TAB** → *Faces*)
5. Select a face on the side of the cone
6. Extrude it about 1/4 of a unit or less (**EKEY**)

7. Merge the extruded face to make the nose pointy (**WKEY** → *Merge* → *At Center*)

8. Enter object mode (**TAB**)

9. Rotate the cone so the nose is aligned with the positive Y axis (**RKEY**, align with the green arrow)

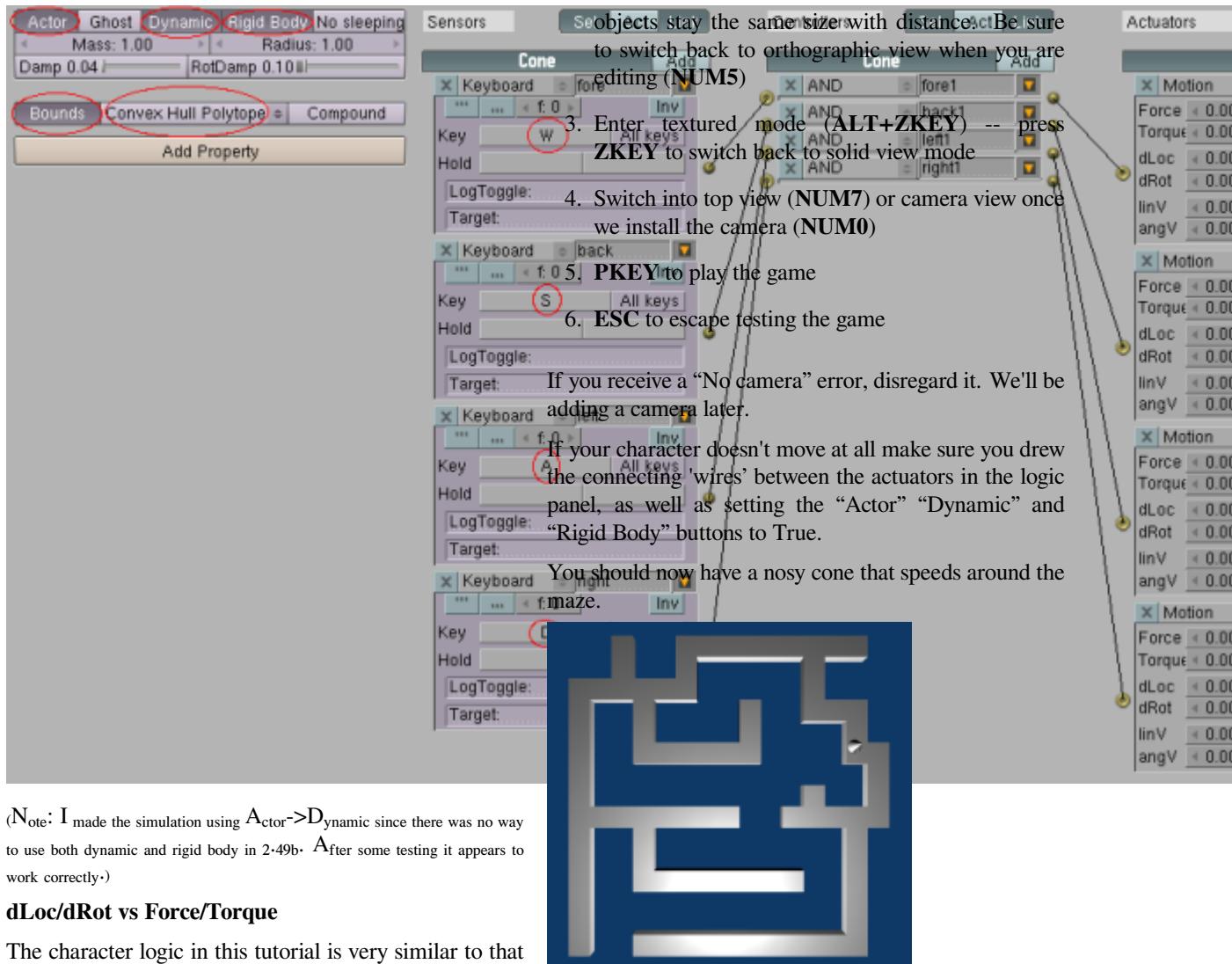
You've got your character now (Not exactly Pixar-like, but it will do for now). Now we just need to give it motivation.

Note: From points 8 to 9 it would seem that the cone should rotated so that the “nose” is aligned with the Y-axis in object mode, but after doing this I encountered the cone still moved in the original Y-axis direction (before rotation). To remedy this I did the alignment of the “nose” to the Y-axis in edit mode instead, this seems to have solved the problem.

3.33.4 Motion

We'll now make the character move, via the use of the WASD keys. In order to do this, game logic needs to be associated with the WASD keys. Copy the configuration seen in the screenshot, taking care to include the convex hull polytope collision in the top left as well as depressing the 'L' buttons, which make the force and torque local to that object rather than global. If the force was global, the protagonist would continue moving in a straight line, even when turning.

1. Go to the Logic buttons (The purple pac-man icon or **F4**)
2. Add four Sensors, Controllers and Actuators.
3. Maximize the logic window (**CTRL+UPKEY**)
4. Copy the configuration seen in the screenshot below.



3.33.5 Testing

Now the time has come for the first test of our game.

1. Add a light source well above the maze (**SPACE** → *Lamp* → *Lamp*), align in front view (**NUM1**)
2. Press **NUM5** to enter perspective viewmode, which gives a realistic view, rather than a view in which

(**Noob Question:** My cone likes to rotate without me pressing **A** or **D**. It also likes to find hidden/nonexistent bumps on the maze surface. How do I fix this?)

(**Answer:** If I understand your question right. If i used torque to turn and force to move forward and pressed forward then turned it started spinning out of control. My solution was to use rotation instead of torque.)

Note: this doesn't work in new versions. The cone just goes through the maze. How do I make the maze block the cone?

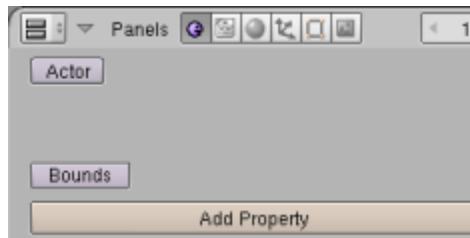
3.33.6 Falling

We will now implement the "death on falling off maze" feature. In order to accomplish this, we'll add something that allows us to check whether the cone has fallen off the maze or not. The simplest way to do this is with a 'skydome' object the entire maze is within, and although there are a number of more elegant ways to accomplish the goal, we will use the skydome method for its simplicity to replicate. From now on we'll need to test the maze with both layers one and two enabled (**SHIFT+2KEY**)

1. Add a material to your cone called coneMat (**F5**, press “Add New”, Links and Pipeline, Link to Object, MA:coneMat).
2. Add a new cube
3. resize it to gargantuan proportions (**SKEY**, just large enough to encompass the maze, and all within it)
4. Move it to layer 2 (**MKEY, 2KEY, Enter**)

Now that you have a massive cube, go to the logic panel and add one Sensor, Controller, Actuator set, and set them as seen in the screenshot.

1. Sensor: Touch, coneMat
2. Controller: And
3. Actuator: Scene, restart



Enable the actor button and press “ghost” this will let the dynamic obstacles, which are added later on, fall through and not get stuck there!

If you test your game now, you should see the game be restarted when you fall off the maze. As we discussed earlier, faces are one-sided in the game engine, so you should not be able to see the cube.

Newbie Question 1: When I did this, my cone just kept on falling. When I moved the cube back to layer 1, it worked fine. Am I missing something? (newb note: ok i had the same problem everytime i was just in layer 1 but when i was just in 1 then i just kept falling)

(Answer: The tutorial says to create the cube in layer 2, although the layer shouldn't matter. Make sure you test the game with both layers one **and** two selected. Also double check the logic for the 'skycube')

(Answer2: I had the same problem. I fixed this by replacing the cube with a grid, scaled and placed under the maze. The grid had the touch logic mentioned above.)

(Answer3: Same here. I thought I'd remove “coneMat” from the material reference, but that wouldn't do if the bricks fell down and touched the cube, it'd just restart the level... the solution: create a property in the cone, a bool value named “isCone” (=true), then in the cube add a sensor ‘collision’ that checks for the object's property ‘isCone’ and keep the rest as shown.)

*(Answer4 (to restart when block hits cube):*Do all as the pic shows you and RMB the cone and go to links and

pipelines and rename the MA: to cone or whatever you want, then RMB the cube then in the Touch sensor then rename the MA: to whatever you called the other MA: . :D this is in 24.9b)

(Answer5: For blender 2.57a you need to place a plane or something under the maze. Then give it (touch -> and -> Scene(restart)) game logic. If it isn't working make sure that actor is selected under physics tab.)

(Question 2: How do I make the box translucent? Right now all I see, also when testing, is the skybox. Also, when I put the box underneath the maze, it works fine, but then none of the objects have shading, while when not showing layer 2 everything looks shaded. Can anybody give a solution? This is in 2.48a)

(Answer: To make something transparent- press **F5**; press the “Material Buttons” button; under the Links and Pipeline tab press Halo and ZTransp, unpress Shadbuf; under Material make A 0.000.)

(Answer 2: My solution was to press the Invisible button (Logic section, next to Ghost in 2.49))

[note: in 2.7X the touch sensor seems to have gone, my workaround was to use a collision sensor, AND controller, and scene actuator with mode: restart and added no game property]

I have put in a new way to die. Add a static Plane, give it a property called Killer and place it just below the level. Add an ALWAYS sensor and connect it to a MOTION actuator that has 0.01 as the top last value. This way it will sloooowly come up. Now, select the player and add a COLLISION sensor. For the property, add Killer. Connect it to a GAME actuator, select Start New Game. You've just added some tension to your game! If you run out of time, you will touch the floor and restart! If you need more time, just put another 0 in 0.01 so it will look like 0.001 Sorry if you can't understand it very well, this is the first time I edited a page on Wikipedia + I don't speak English very well. Have fun! :)

3.33.7 Camera

SPACE → *Camera* near the protagonist cone. There are two easy ways to do the camera: logic camera and child camera. In this case you might be better off with the logic camera, but it really is a matter of personal preference.

- **Logic Camera:** Add to the camera's logic panel

1. Sensor: Always
2. Controller: And
3. Actuator: Camera, to object “Cone” (or whatever your protagonist is called--look under the object panel when you have the protagonist selected) height 5, min 5, max 10.

- **Child Camera:** Select the camera, then select the protagonist (order is important) then

CTRL+PKEY to parent the cone to the camera. Align the camera in a view that you like, and test the game.

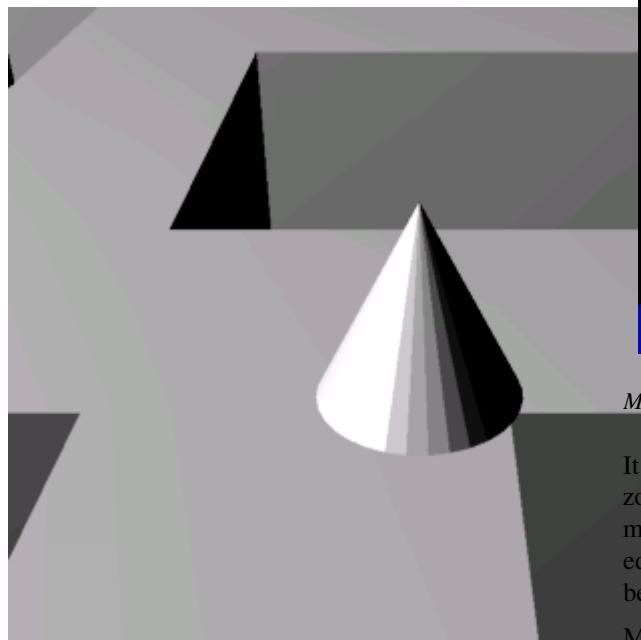
- **First Person Camera:** Well, this one doesn't really count as another method, but if you're willing to redo your motion to just move the camera instead of the character, (just do everything you did for the character for the camera) you can run around in the first person. Or you can just put the camera in the cone because one of the sides always are invisible (look in texture mode) and parent it to the cone. REMEMBER: To access the ingame camera during gameplay, before you start, hit ctrl 0 or you can go to (view > camera)

(Newb Question 1: No matter what I do (even delete the camera!), when I hit 0 or ctrl+0, I always see either only the death plane (I used a plane below the maze instead of a cube around it) or nothing, depending if I have layer 2 selected or not. There's only the one camera (or none, if I delete it), according to the Ops Schematic. What's wrong?)

(Answer: Ctrl+0 assigns the selected object as a camera. To solve your problem select your camera and press ctrl+0, to enter camera view press 0)

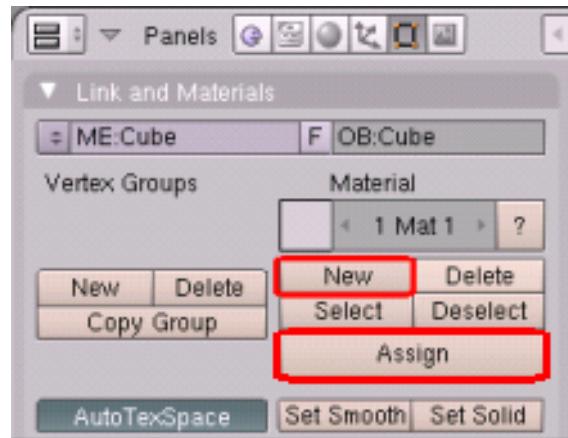
3.33.8 Beautification

Blender is a great 3D modeling program, and with such a vast number of tools at your disposal, you should be able to make your game look better than this:



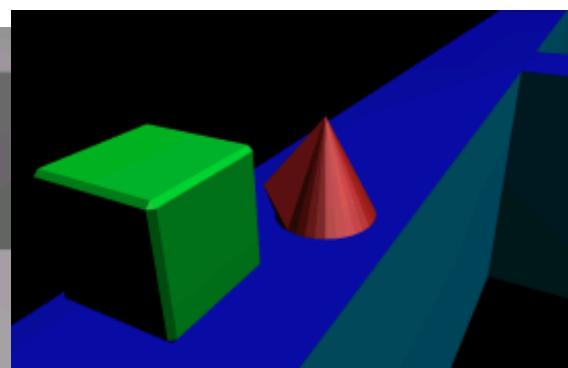
We'll add one or two textures to each of our objects.

1. Select the object you want to texture
2. Go into edit mode if you want your object to have more than one color



Press the 'new' button then the 'assign' button to assign a material to a number of faces

3. Select all the faces you are going to assign to a color (**CTRL+TAB → Faces**)
4. Go to the edit panel (**F9**) on the buttons window
5. Near the left there should be a panel that says “Link and Materials”. Click on the buttons highlighted in the screenshot
6. Adjust the color of that material by going to the material buttons and changing the “Col” (color) value to something other than white

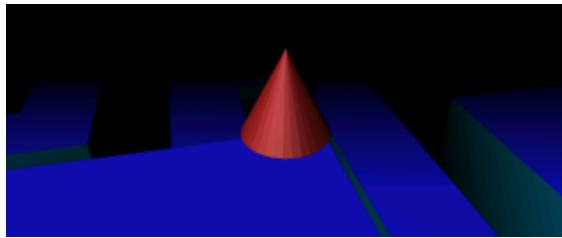


Maze game with materials

It looks better now, but when the camera is nearly horizontal, you can see where the camera stops rendering the maze. You can fix this by selecting the camera and, in its edit buttons, changing its clipping range... but there is a better way. Mist.

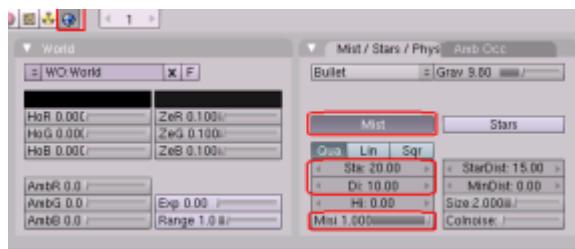
Mist obscures everything a certain distance away, is the same color as the world texture, and can be handy. We'll add some mist to our game so the player can't see too far ahead of him, thus making the game too easy.

We'll make the mist quite close, although you can vary the distance. Go to the Material button, and to the World sub-buttons, then copy the settings in the screenshot. When



With mist and raised sections

you test this out, make sure you are in texture mode to see the mist.



Question: I copied the screenshot but i can't get the mist to work. What's going on?

(Answer: Change Start to a smaller value like 15.)

Question: Is the screenshot correct, or are my sizes different. I found that a mist distance of 10 was barely visible, yet a distance of 1 had a much better effect. Is this normal?

Question: I have been able to create the mist, and can see it when I start the game, but the background colour is still grey, so the maze is obscured but everywhere else is still grey.

3.33.9 Levels

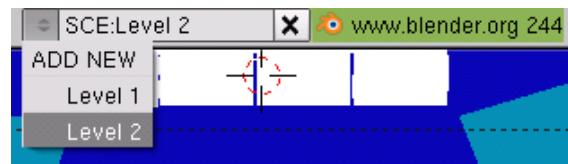
Now all that remains is to create the next level with some obstacles.

1. Add a cube at the end of the maze, just above the floor level so the cone can touch it (**SPACE** → **Add** → **Mesh** → **Cube**)
 2. Press the Scene dropdown menu and select “ADD NEW” → **Full copy**
 3. You can now go back to the Maze Surface section to make a new maze mesh
 4. Go back to level one via the Scene menu
 5. Select the End of level cube
 6. Change its logic panel to resemble this:
- Sensor: Touch, coneMat
 - Controller: And
 - Actuator: Scene, Set Scene, Scene.001 (or whatever you named your level 2)

(noob: When I went into level 2 my player name changed to coneMat.001 and I had to change all my goals and the giant restart cube assignments.) *Answer:* Rename the coneMat.001 to whatever you had the material named for the first scene - this fixed it for me.

(noob: In my level 2 my cone hits the cube and nothing happens!) *Answer:* Make sure that the Sensors, Controllers and Actuators tabs under the Logic panel (F4) are connected with wires.

(noob: When I went to level two I can't see anything!)



The scene button is on the top panel

To make it so multiple objects have to be touched (or acquired by the character, as it may be), give each object the logic settings described above and change the actuator type to Message with the subject “goalget”. If you'd like the object to disappear after it is collected, add an Edit Object actuator to the object of the type “End Object”, and link it to the existing AND controller.

Next, select the massive cube that restarts the level upon contact with the character. If you chose not to include one such cube, create an empty at the start of the level (so you can easily find it again later). Give the cube or empty the following logical operators:

- Sensor: Message, goalget
- Controller: And
- Actuator: Property, Add, wincon, -1
- Sensor: Property, Equal, wincon, 0
- Controller: And
- Actuator: Scene, Set Scene, Scene.001 (or whatever you named your level 2)

Then click “Add Property” and give it the Type Int, the Name wincon, and a value equal to the number of objects needed to be collected to complete the level (if all of them, make it equal to the number contained in the level). This can be expanded to include other multiple win conditions besides the collection of an item by making the completion of each condition subtract from “wincon”.

You now have a multilevel maze game. Feel free to elaborate on the gameplay mechanics and models so they look better.

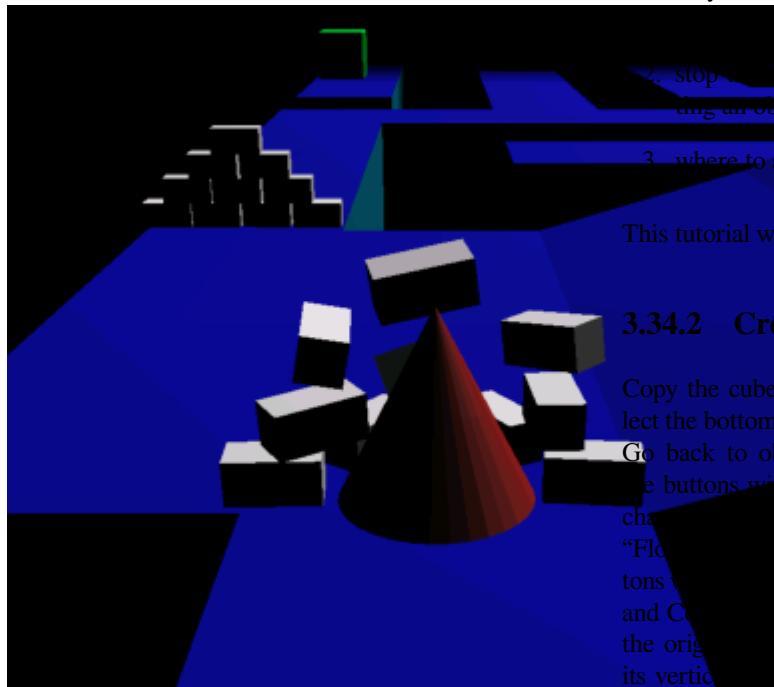
Noob note: I typed in where i thought everything should go but it's not doing anything. I'm not sure of what I'm looking for while in playmode.

3.33.10 Dynamic obstacles

This part of the tutorial will assume increased skills and knowledge of the Blender interface and basic commands. As such, this section will be written less specifically, and the shortcuts for all basic commands will not be covered here. From this point on, this tutorial is of an intermediate level. If you cannot follow this part of the tutorial, you can learn the basic functions used below in an earlier tutorial.

We will now add some dynamic obstacles. In this case, they will constitute a 'brick' wall that can be broken through by your character.

1. Add a cube
2. Scale it (in edit mode) to the dimensions 1,0.5,0.5
3. Bevel it (WKEY) recursion 1, size 0.03 (press space to enter value manually)
4. In logic panel, set it to Actor, Dynamic, Rigid body, Mass 0.5
5. Go right below that to click Bounds and leave it on "Box". That makes Blender realize that what you added is a box, and should act like it (instead of rolling, it will slide now)
6. Move it to an intersection in your maze (GKEY) and duplicate it until you have a brick wall (ALT DKEY, LMB)



You now have a barrier that your character can break through.

By using this

3.33.11 Conclusion

You have now learned enough of the Blender Game Engine (BGE) to create your own game. If you wish to go very far into Blender games, I recommend you learn Python, as trying to make a full game within the graphical interface for the BGE is like trying to dig a grave with the blunt half of a toothpick. Blindfolded. And the toothpick is glued to your forehead. However, you are now well on your way to becoming a game-maker! For a basic beginner tutorial about Python, "[Open Source Video Game](#)" series.

If you have any questions, concerns, or 'noob notes', please post them in the discussion page for this tutorial, and not in the tutorial itself.

3.34 Platformer: Improving the Physics

3.34.1 Introduction

This tutorial is intended as an improvement on its prequel, [Platformer: Creation and Controls](#), and will require files created through that tutorial. It is also recommended that you read [An aMAZEing game engine tutorial](#) as well before beginning this tutorial. It will require a familiarity with the Blender UI, simple commands (such as AKEY to select) and basic modeling skills. This tutorial will introduce the following features and improvements:

1. only letting the character jump while touching the ground
2. stop the character from bouncing greatly upon hitting an object
3. where to adjust material friction

This tutorial was written for version 2.45.

3.34.2 Creating Your Hit Test Object

Copy the cube with **SHIFT-D**. Go into edit mode, select the bottom four vertices of the cube and delete them. Go back to object mode. Go to the editing panel in the buttons window. In the Link and Materials section, change the ME: value to "Square" and the OB: value to "Floor". In the Mesh section, there will be three buttons with the word "center" in them; Center, Center New, and Center Cursor. Press Center New. This will change the origin of the "Square" mesh to the center of all of its vertices instead of the center of the cube you made it from. Scale it to 0.99% of the original and move it up 0.01 along Z. Now go to the shading panel and make sure you're in the Material buttons subpanel. Find Links and Pipeline, and make sure "ME is selected. Click the

X next to the material selection (under the words “Link to Object”). Then select “OB” and press “Add New” and name this material “FloorHit” (again, no quotes). Make it green colored, so you can easily find it while editing, but turn on “Shadeless” and “No Mist” in the Material section and “OnlyCast” in the Links and Pipeline - Render Pipeline section and turn off “Radio”, “Traceable”, and “Shad(ow)buffer” in the Render Pipeline and “Shadow” in the Shaders section. This will make it completely invisible at runtime and take up little resources.

(question: it wont make it invisible it just flashes rapidly)

Making the sensor require floor contact

Now select Suzanne and go to the logic panel. Add a touch sensor called “jumpcol” (for ‘jump collision’) and connect it to the same AND controller as the jump keyboard sensor. Set the f to 10 and the MA: to FloorHit. This will make it so your character can only jump while it’s touching something with the FloorHit material. By that same token you can link this to *all* of your movement-related AND controllers so that the player can’t adjust it’s movements in mid-air. This is a nice physics touch but for most platformer games, where mid-air dexterity is almost essential, it doesn’t work. You also have to use Force and not dLoc, otherwise your character won’t be able to move at all while jumping, unless your character only needs to hit things with his/her head, and not jump over gaps. Note that this still does not keep the character from jumping while touching the hit test object at the sides, so they could still jump if they were touching the side of the collision surface, but at least they can’t jump in mid-air or while touching the bottom of the ground. I have yet to find a solution to this problem not involving Python.

3.34.3 Excessive Bouncing

Another problem with the original model, as you might have noticed, is that when you run into one of the ground cubes you bounce back a great deal. To fix this, turn to the materials panel with one of your green hit test squares selected. Under the color selectors there should be three buttons that say “RGB”, “HSV”, and “DYN”. Select DYN, and turn the restitution most or all the way up. As you can see, you can also find the friction property from here, if you want your ground to be more or less frictional (like mud or ice).

3.34.4 Final Notes

Always copy your floors, their respective hit test objects, and other generic objects with **ALT-D** and not **SHIFT-D**. This keeps you from duplicating things that remain the same for every one, like the material and mesh, which quickly drain resources if not recycled. Since there’s not

much further you can take the Platformer without learning Python, I suggest you start looking up information on the Blender API. Some essential information is to be found in the [procedural object creation](#) tutorial. A Blend file of the tutorial’s finished product is coming soon.

Note: A (possibly) simpler method for jump limitation can be found on the discussion page for this tutorial.

3.35 How to Make an Executable

This tutorial will show you how to make an executable for your game made in Blender.

Note: The methods listed in the “Windows” and “GNU/Linux or Mac OS X” sections work only for the operating system you are on when you create the file. To make it cross platform, use the “BlenderPlayer” method.

3.35.1 Windows

First create a folder that will hold all your game information. Name it something meaningful, like “Yo Frankie!”.

The folder must contain four files that you can copy from your blender installation directory.(under windows it should be “C:\Program Files\Blender Foundation\Blender”) They are:

1. SDL.dll
2. python24.dll (*Or other relevant python file; python25.dll, python26.dll etc.*)
3. pthreadVC2.dll
4. zlib.dll

(sometimes you will need the following also...

1. avformat-51.dll
2. avutil-49.dll
3. avcodec-51.dll

They are in the same folder.)

Not quite a Noob Note: If you are using the latest version of Blender you WILL need the following files)

I am running windows xp media edition and blender 2.46, here is a full list of files i needed:

1. avcodec-51.dll
2. avformat-52.dll
3. avutil-49.dll
4. libfaac-0.dll

5. libfaad-0.dll
6. libmp3lame-0.dll
7. libx264-59.dll
8. pthreadVC2.dll
9. python25.dll
10. SDL.dll
11. swscale-0.dll
12. vcomp90.dll
13. xvidcore.dll
14. zlib.dll

On Blender 2.48a (Windows XP Media Center 2002, SP3), these .manifest files are also required:

1. blender.exe.manifest
2. blenderplayer.exe.manifest
3. Microsoft.VC90.CRT.manifest
4. Microsoft.VC90.OpenMP.manifest

If you use *random* Python module in your game, you will have to add one more file to your game directory. For Blender 2.49 it would be python26.zip, which you can find in Blender main directory (where blender.exe is). Otherwise there may be some errors during executing the game in system without installed Python.

Since Blender 2.56 you will need to enable Save As Runtime; first open blender with the game that you have created and open the file menu. Click on User Preferences then select Add Ons then Game Engine, check the box Game Engine Save As Runtime and return to the file menu. Save As Runtime will appear as an option in the export menu then save to the new folder that you have created and rename the file yourgamenename.exe, and then you can run the game!

If you encounter errors during the “Save As Runtime” process (I’m on Windows 7) you can right click on the Blender icon and select “Run as administrator”. So you will have your fresh .exe and you can spread it with a piece of your favourite pepperoni pizza.

Making A Screensaver (Windows Only)

1. First, save your runtime (that’s the .exe addressed above.) via File->Save Game as Runtime. (“Save Runtime” in older versions)

2. Now, go to your .exe game and rename it .scr - for instance, if your game was NotMyGame.exe, rename it to NotMyGame.scr. You can now right click to Install it, and then use it as your regular screensaver by applying it as you would any other screensaver (right click on desktop, properties... You know the drill.)

Screensavers are not games, and so they should not accept input. At most, they should be videos showing what your game does. If you just rename your regular game as a .scr, it will be remarkably boring, because your game needs input and screensavers do not.

Intel screensaver bug

There is a caveat with Intel Integrated Graphics drivers, found in many laptops though. The graphics driver shuts off OpenGL acceleration for screensavers, for some obscure reason. The way to work around this is to rename the .scr extension to a .sCr as the driver’s algorithm is dependent on case-sensitive characters. If you experience very low framerates in your screensaver, you should attempt this fix, It is tested and reported to work.

3.35.2 GNU/Linux or Mac OS X

Open blender with the game that you have created and open the file menu. Click on **Save Game as Runtime** and then save to the new folder that you have created and rename the file yourgamenename...

and then you can run the game!

3.35.3 BlenderPlayer

The methods shown above only create an executable for your operating system. Well, BlenderPlayer can fix that.

1. Make a new folder to store all your game data.
2. Then save your .blend file into the directory.
3. You can skip this step and the next step if you do not want a Windows version. For the Windows users, copy blenderplayer.exe to the new folder from a Windows copy of Blender. Then copy all your DLL files for Blender as mentioned for Windows to the folder.
4. Next you have to make an MS-DOS batch file (for UNIX users, this is the shell script equivalent). In a simple text editor, in CR-LF mode if available (Notepad is always in this mode, and NOT a word processor!), copy and paste this text:

```
blenderplayer.exe yourgamenename.blend
```

Save it as YourGameName-Windows.bat in your game folder.

5. You can skip this step if you do not want a UNIX (basically Mac OS X and GNU/Linux) port. For GNU/Linux (at least), make a shell script. (A shell script is the UNIX term for a batch file.) In a simple text editor, in LF mode (unfortunately Notepad can't be used), copy and paste this text:

```
#!/bin/bash
./blenderplayer.app/Contents/MacOS/
blenderplayer yourgamenname.blend
Save the file as YourGameName-UNIX.sh in your
game folder. You will need to have BlenderPlayer
in the same directory as the .blend game file.
```

6. Write a readme for your program. This is again best done with a simple text editor like Notepad or gedit, but it does not matter which mode it is in. You should include the name of the game, a description, perhaps a walkthrough or hints, and if you made a *NIX port, mention that it requires BlenderPlayer, available with Blender.

3.35.4 Legal Issues

Blender and the BlenderPlayer fall under the GNU General Public License. Blend files are copyrighted to their respected owners and do not fall under the GPL as long as they are not packed in the BlenderPlayer. If a user does not want the blend files to fall under the GPL, it is recommended not to use the “Save Game as Runtime” feature. To put it another way, the user must keep the blend files and the BlenderPlayer in separate files. More information can be found at <http://www.blender.org/education-help/faq/gpl-for-artists/>. (Blender 2.4 now has a run game from file actuator.)

3.35.5 Proprietary Blend Files

Blender has no built-in functionality to “lock” or protect user generated content. As a result, anyone who has Blender can open and/or modify blend files. However, it's still possible to lock blend files. Common methods involve encrypting the blend file and then temporary decrypting it at runtime. This can be accomplished by using python scripts or by using external 3rd party applications (which are feasible under the GPL v2).

3.36 Build a Skybox

3.36.1 Prologue

One way to add a realistic feeling to your 3d environment in a game engine is to create a skybox. A skybox is a large cube which has on its inside a projection of a 360° environment. When the player (camera) is inside this environment, the scene is rendered with the illusion of being

inside a gigantic world. This is a similar effect to Quicktime VR (see <http://www.fullscreen360.com/> for examples). And, by setting up the skybox as a simple cube shape, you place the least amount of strain on the graphics engine. It's a great advantage for your game with very little overhead.

This tutorial will show you how to create skyboxes relatively easily from panoramic photos. My favorite part is, you can do it easily using free tools such as Blender and the Gimp.

Using the Gimp to manipulate images is not really in the scope of this tutorial... check out some other page on using that software. You should have an understanding of how to edit images and apply alpha channels. (You could also use the Gimp to apply a polar coordinate texture to your rectangular image in order to create a fisheye image. Hint: it's not the sphereize filter.)

3.36.2 Gather your graphics

You can take panoramic images yourself using a regular digital camera and a tripod. A quick way to accomplish this is to draw marks on your tripod base at every 30 degrees (think of the hours on a clock face). Make a single mark on the swivel of your tripod to allow you to line up your shots -- twelve shots at 30 degrees each. Then, using a program such as [the Gimp](#) or the incredibly cool [Autostitch](#) to merge the photos into one big panorama.

Or, if you're lazy like me, you can just grab photos online to use as templates to create original images. There are also many places you can download non-copyrighted photos for free as well. One resource for cloudy sky textures, as well as panoramic photography instructions, is Philippe Hurbain's site [Philo's Home Page](#). This tutorial will use a fisheye sky photo from his copyright-free Panoramic Skies images collection.

You'll also probably want a photo for your ground, unless you prefer to use real models such as buildings in your skybox. This [earlier chapter](#) on creating landscapes can be incorporated into setting up your skybox. However, this tutorial will use the sky photo for the top half of our world, and a panoramic landscape with an alpha channel for the bottom half. I've created a ground image using copyright-free textures obtained from [Accustudio](#).

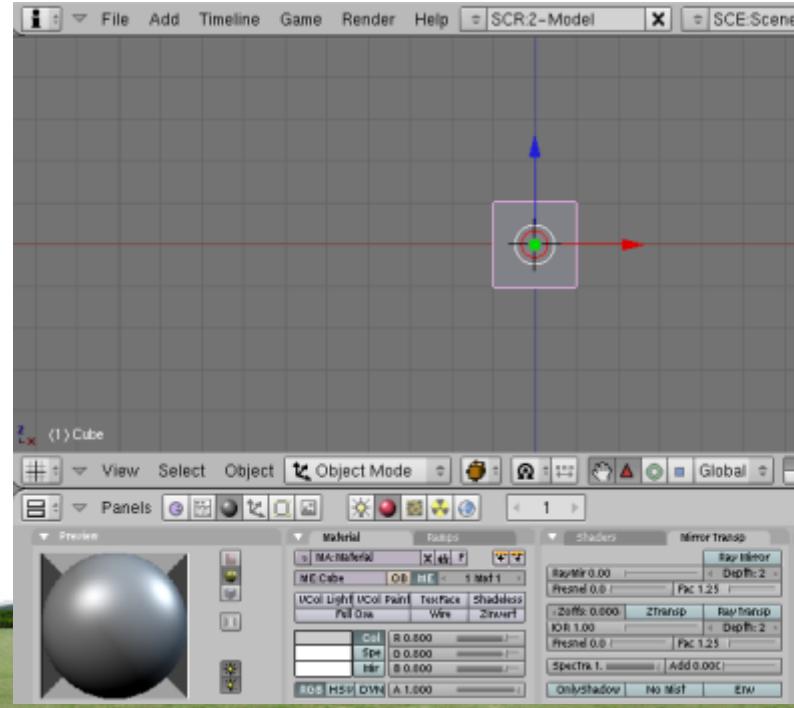
Here are the images I'll be using (you'll want to use images with higher resolution): Note that the sky has trees, etc.



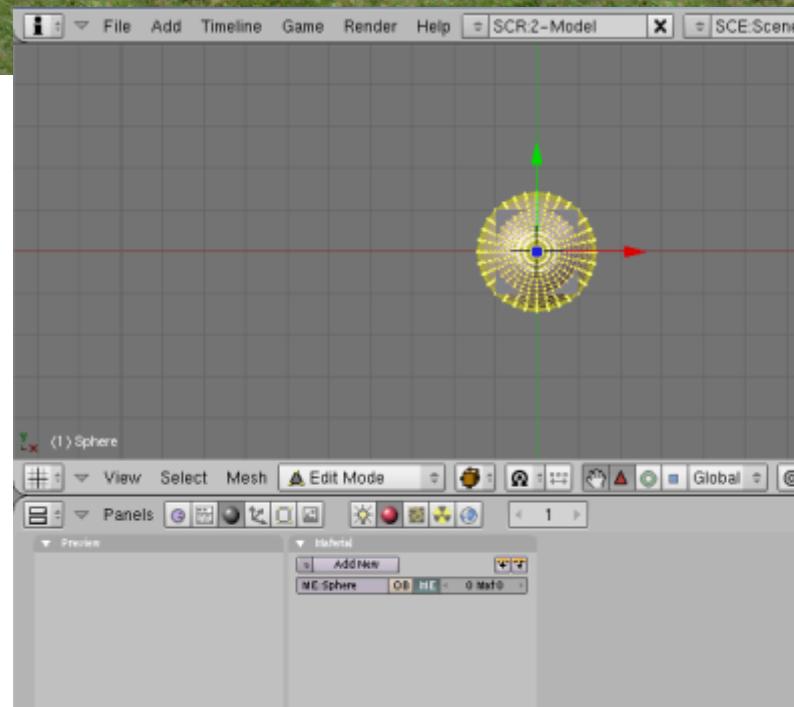
Note: I've outlined the horizon of the ground texture with an alpha channel which will allow me to place the ground mesh right against the sky mesh with a very natural feel.

3.36.3 Create a dome for the sky

Open a new file in Blender. Your default new file will probably be a two-unit cube in the center of the screen, with a single light source and a camera. You can delete the light source because we won't be needing it. Leave the cube, because that is what will become our skybox.

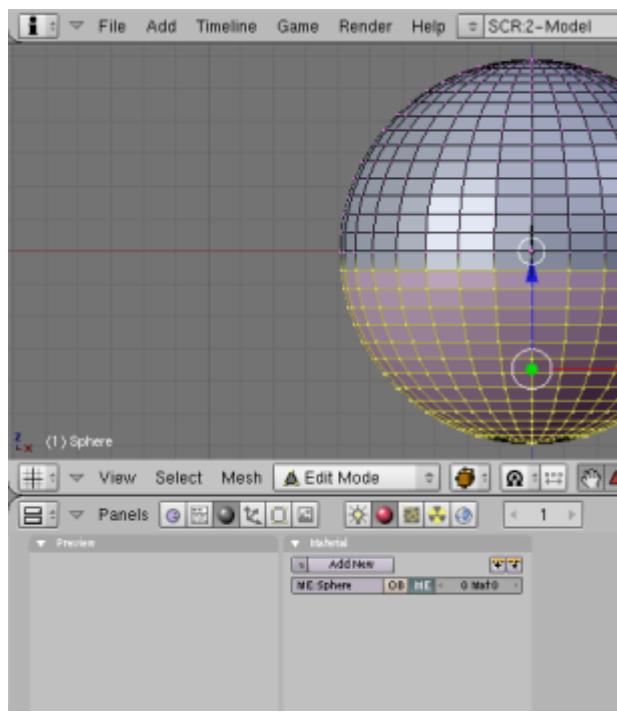


The cube will be the center of our environment, so use Object->Snap->Cursor To Selection if your cursor is not centered. Then, from the top view [KEYPAD-7], Use [KEY-SPACEBAR] to insert a new mesh; make it a UV sphere. I find a 32-segment, 32-ring sphere to be sufficient. We create the sphere from the top view because that is the projection from which we want to add the sky texture.



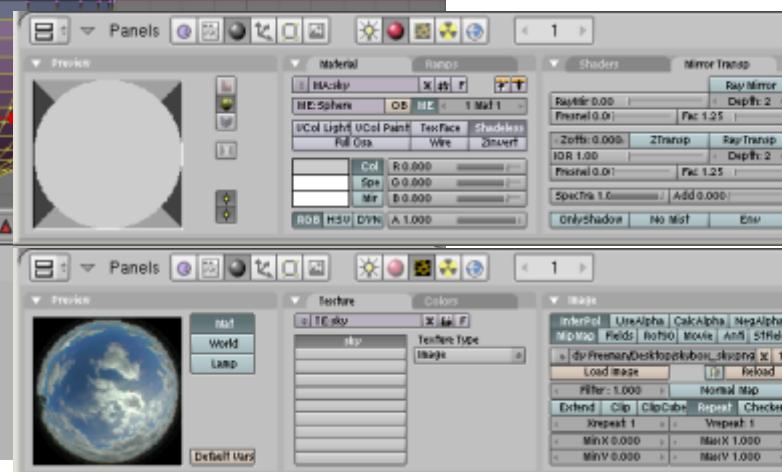
Scale up the sphere so it resembles a large “arena” in comparison to your cube, and select and delete the lower half of the vertices, using the front view [KEYPAD-1] and [KEY-B] to create a bounding box. It helps if “Select Visible” is turned off so you can select all of the vertices

in one go.

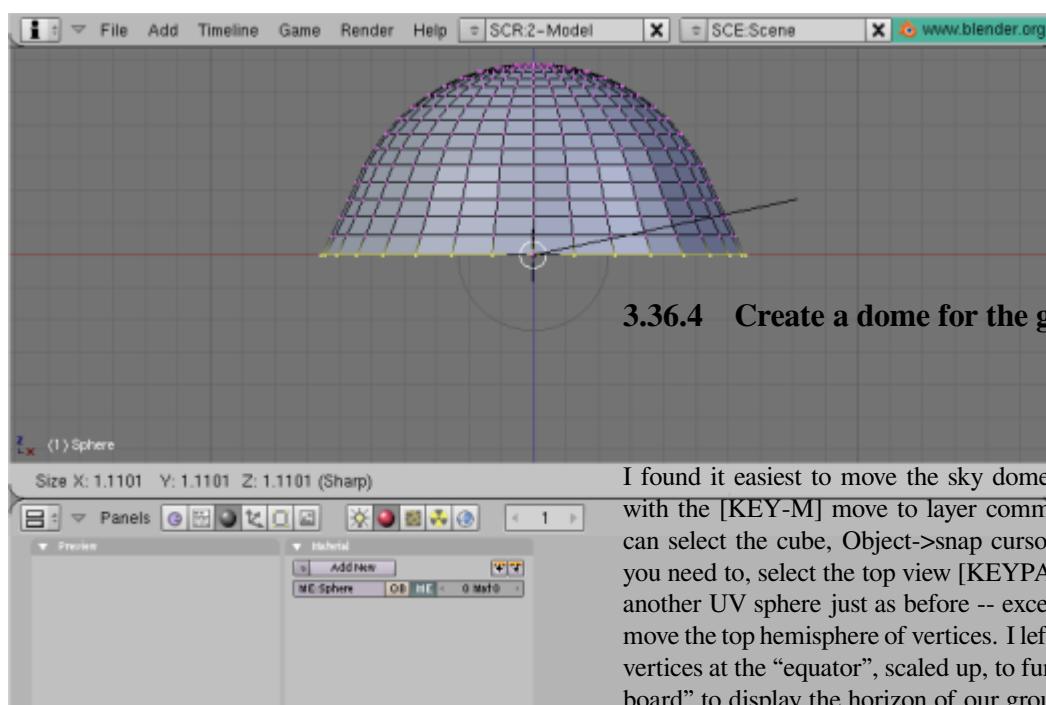


Turn on proportional editing with [KEY-O], then select the bottom row of vertices and scale them up with [KEY-S] so that the bottom of the sphere gets a bell shape. Because the projection of the sky texture will be from the Y-axis (ceiling) we need the bottom faces of the sphere to be at an angle, to catch the texture. (Faces perpendicular to the projection will look like smears.) Alter the influence of proportional editing with [KEY-PAGEUP] and [KEY-PAGEDOWN]. Linear or Sharp falloff works best with the sphere shape.

In the Materials menu, create a new material and a new texture. Be sure to set your material not to receive shadows by clicking the “Shadeless” button. Then, in the Texture menu, set the texture type to Image, and click the Load Image button to insert our sky texture. Back in the Materials->Texture->Map Input menu, you may need to scale your image to get rid of the distorted textures at the edges of the fisheye by setting the Size to, say, 0.950 for X, Y and Z.



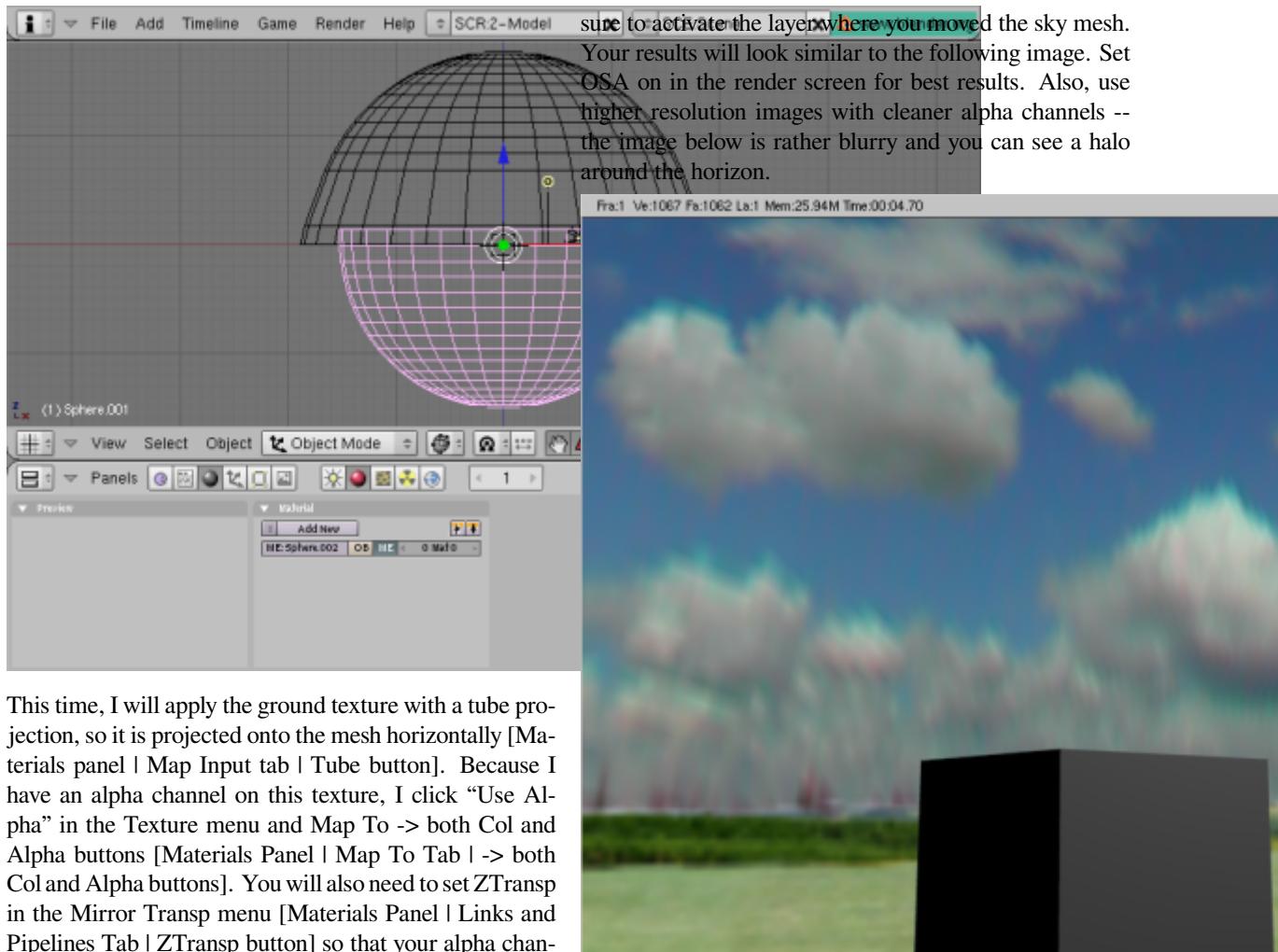
At this point, if you wish, you can reposition the camera and render the scene to see how your sky mesh looks.



Now you're ready to add your sky texture to this mesh.

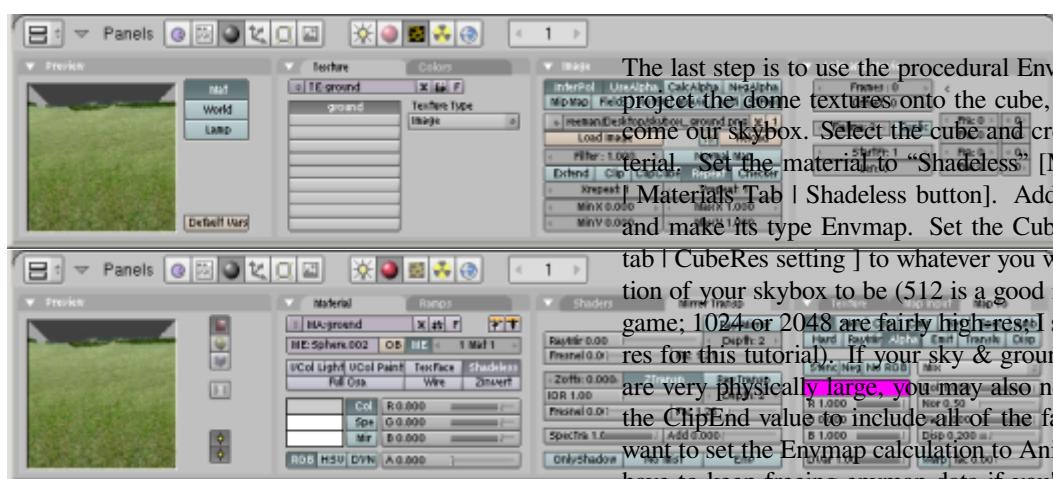
3.36.4 Create a dome for the ground

I found it easiest to move the sky dome to a new layer with the [KEY-M] move to layer command. Then you can select the cube, Object->snap cursor to selection if you need to, select the top view [KEYPAD-7] and insert another UV sphere just as before -- except this time, remove the top hemisphere of vertices. I left an extra row of vertices at the “equator”, scaled up, to function as a “billboard” to display the horizon of our ground texture with the alpha channel. This sphere should be slightly smaller than the sky hemisphere.



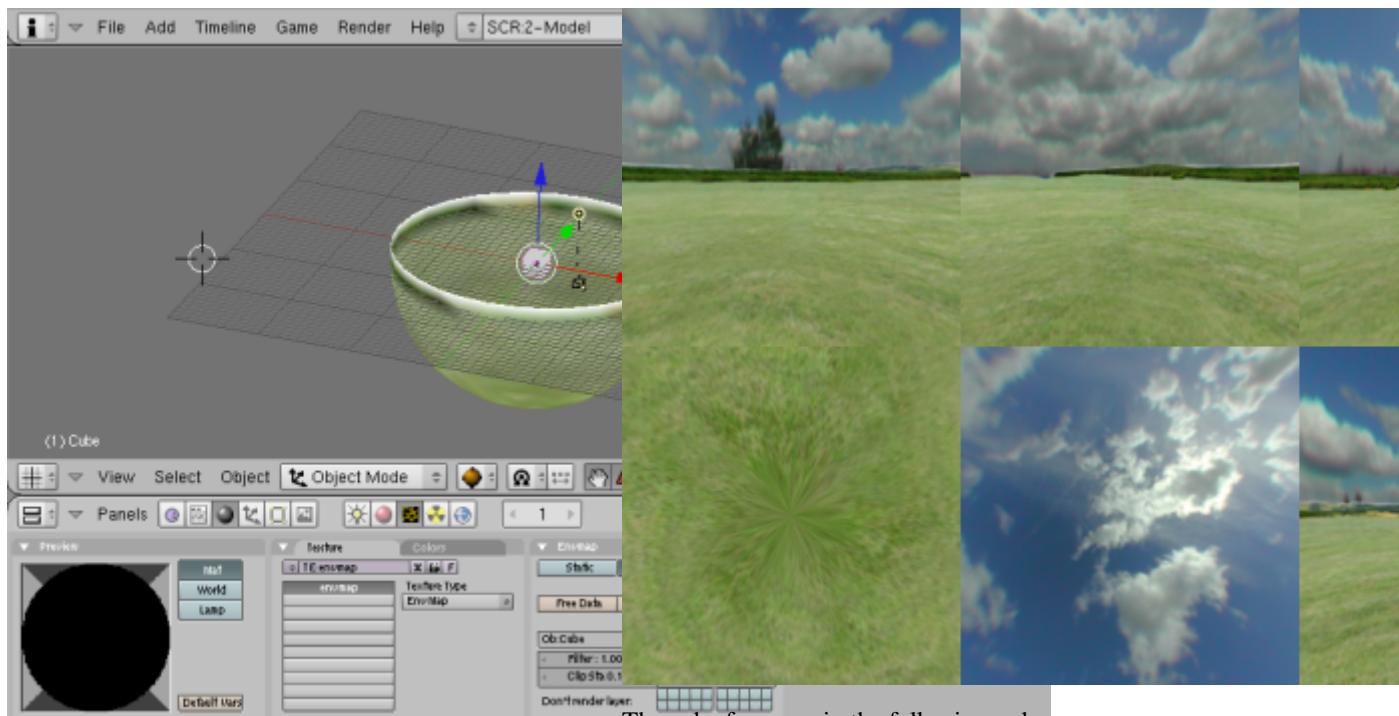
This time, I will apply the ground texture with a tube projection, so it is projected onto the mesh horizontally [Materials panel | Map Input tab | Tube button]. Because I have an alpha channel on this texture, I click “Use Alpha” in the Texture menu and Map To -> both Col and Alpha buttons [Materials Panel | Map To Tab | -> both Col and Alpha buttons]. You will also need to set ZTransp in the Mirror Transp menu [Materials Panel | Links and Pipelines Tab | ZTransp button] so that your alpha channel shows up in the envmap (which will become your skybox), and Alpha to 0 [Materials panel | Material tab | A slider] to allow the masked areas to be transparent. (Alpha channels appear to require Z buffering to appear on procedural textures.) Also, you may need to adjust the offset of the ground texture (Y-axis), so that the horizon appears properly on the “billboard” area of your ground hemisphere.

3.36.5 Render the environment map



Again, you can reposition the camera and render the scene to make sure everything is properly aligned. Be

The last step is to use the procedural Envmap texture to project the dome textures onto the cube, which will become our skybox. Select the cube and create a new material. Set the material to “Shadeless” [Materials Panel | Materials Tab | Shadeless button]. Add a new texture and make its type Envmap. Set the CubeRes [Envmap tab | CubeRes setting] to whatever you want the resolution of your skybox to be (512 is a good resolution for a game; 1024 or 2048 are fairly high-res; I stuck with low-res for this tutorial). If your sky & ground hemispheres are very physically large, you may also need to increase the ClipEnd value to include all of the faces. You may want to set the Envmap calculation to Anim so you don't have to keep freeing envmap data if you're experimenting. (Anim automatically clears Envmap data with every render, otherwise you must click 'Free Data' to reset the Envmap.)

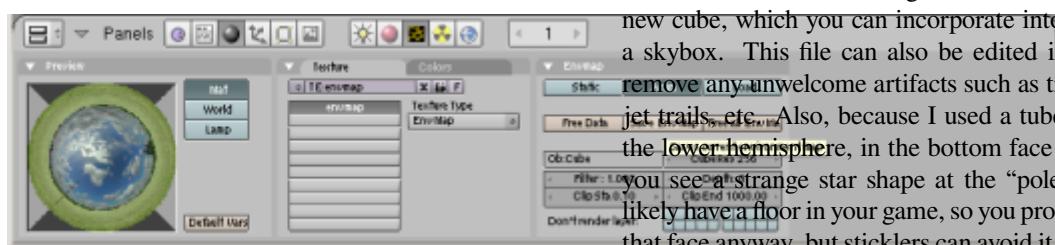


The cube faces are in the following order.

LEFT (-X)	BACK (-Z)	RIGHT (+X)
BOTTOM (-Y)	TOP (+Y)	FRONT (+Z)

Once you've created the Envmap texture, you should be ready to render the Envmap for your skybox. If you want to set your file format such as JPG or PNG, you should do that first. Then, simply go to the render screen and click "Render." Again, make sure all layers are visible. The rendering window appears. First, Blender renders the environment map of the cube. Afterward, the camera view is rendered, at which point you can hit [KEY-ESCAPE] to stop rendering -- we are only interested in the environment map which is already complete.

Select the cube again, then get to its texture menu. You will see the newly-rendered Envmap on the sample texture. Click "Save EnvMap" in the texture menu to save the rendered Envmap.



Blender environment maps are saved as a 3x2 matrix of squares, as seen here:

You can now load this image as an envmap texture in a new cube, which you can incorporate into your game as a skybox. This file can also be edited in the Gimp to remove any unwelcome artifacts such as trees, buildings, jet trails, etc. Also, because I used a tube projection on the lower hemisphere, in the bottom face of the envmap you see a strange star shape at the "pole." You'll most likely have a floor in your game, so you probably won't see that face anyway, but sticklers can avoid it with clever use of the Filters->Distorts->Polar Coords filter in the Gimp or Filter->Distort->Polar Coordinates (Polar to Rect.) in Photoshop. Patching also works well.

To make the skybox appear as a static background in your game, vertex-parent it to the current active camera object.

3.36.6 Video Tutorial

Ira Krakow's Blender 2.49 Skybox Tutorial: <http://www.youtube.com/watch?v=azkk3JrM5Es>

3.37 Basic Mouse Pointer

3.37.1 Mouse Pointer

Making a simple mouse pointer in the game engine. This takes up a lot of resources but it is very simple.

A little Python is involved but it is very easy to use and is only 2 lines of code.

1. Open up blender and split the screen in two.
2. Make the right screen a text editor and add a new text file with **ALT+N**. Type in the following code.
import Rasterizer as r
r.showMouse(1)
3. Set *TX: showpointer* in the middle of the Text panel menu bar.
4. Select an object that will always be available - preferably a camera.
5. Go to the logic tab, add a “*Property*” sensor, a “*Python*” controller, an “*AND*” controller and a “*Property*” actuator.
6. Activate the *True Level Triggering* (the ““ button), set *Prop: switch* and *Value: 0*. Connect the property sensor to the python and AND controllers by dragging lines between the bullets.
7. For the python controller set *Script: showpointer*. Note: If the value keeps being reverted to blank after setting it, the name you entered is not a legal script name; chances are you did not set the name of your script correctly. Look for the selection menu beginning with “TX:” and make sure it says *TX:showpointer*.
8. Now connect the AND controller to the property actuator. Set *Prop: switch* and *Value: 1*.
9. Select *Actor* and click on *Add Property* make it a *Int* type and set *Name:switch*.

Now press **P** to start the game and now you'll see your mouse pointer.

3.38 Text in BGE

3.38.1 Text

There are a lot of tutorials that show how to make text for the Blender Game Engine, for example to use in menus. Most involve editors, graphics programs, TGA files, UV mapping, scripting, higher magic and so on. Here is a different approach that takes but a minute.

1. Start Blender.

2. Remove the default cube. (**X**)
3. Add text. (**SPACE** → *Add* → *Text*)
4. You may want to switch to edit mode (**TAB**) and use the variety of features that Blender provides for editing, formatting and laying out text. They are described in the [Blender Manual Section](#). For a start just use **BACKSPACE** to delete the letters “Text” and type your own like “This is simple”.
5. When finished editing go back to object mode. (**TAB**)
6. Convert your text to a mesh. (**ALT+C** → *Mesh*)
7. Press **P**

This is as simple as it gets. Of course you can do all kinds of laying out with your text before the conversion in step 6 (check the [Blender Manual](#)) as well as modeling, texturing and dynamically manipulating thereafter (as you have a normal mesh). Simply remember that you need to make sure the text is final before you convert it.

This tutorial has been tested with Blender 2.8 RC1.

“However, it should be pointed out that while this is so incredibly simple, it is not poly-count efficient. Remember, there is more to making a game than making it look good-making it run well. If you need a cut-and-dry menu, this is the best way. If you need to worry about performance, then it might be worth it to check out some UV Mapping and external program usage.

It is also important to note that these fonts can't be dynamic, contrary to bitmap-uvmapped ones...

3.38.2 Links

- How to create Dynamic Text in Blender Game Engine (Youtube)

3.39 Platformer: Creating the Engine with Python

3.39.1 Introduction

Note: Python code is placed in a Text Editor window. It might be helpful to split your 3D View window into a separate part, so that you can use a buttons, 3D view, and text editor window simultaneously.

The initial stages of the Python Platformer tutorial series will mostly have to do with replicating what was done in the logic board Platformer tutorials. If you have not read and understood those tutorials, you may not understand exactly what the function of most of this code is. This tutorial details:

1. An explanation of the “code sections” to be modified, rewritten, or redefined frequently throughout the series
2. Creating an object linked to a pre-made mesh
3. Linking an object to a scene

and will detail by completion:

1. How to move an object in response to keyboard triggers

3.39.2 Code Sections

We will refer to the sections of the code in this manner throughout the series:

Import Section

The series of import commands at the beginning of the piece of code, like the contents of the head tag in HTML documents. Python has a basic set of commands kept naturally in the language, and the rest are imported so that a large amount of commands aren't loaded when not needed, causing unnecessary memory expenditure. This lends the advantage of being able to extend the language by writing custom sets of commands to be imported into Python, which is how Blender interfaces with Python. For this tutorial, our import section is:

```
import Blender import bpy
```

When you import something as something else, it basically creates a variable equal to the imported module's name, so that you don't have to type it out. It is inessential, but it makes the coding go faster. You might want to run a find and replace search on your document afterwards, replacing all instances of the name you imported it as with the real module's name and deleting the part that imports the module by another name, as this can make the code run slightly faster.

Essential Footer

This is the collection of commands that you must remember to include at the end of the document, regardless of any changes made to the code, for it to work. It must be noted whether you should change the entire footer (as if a feature was removed from the main code-in-progress) or adding a command to your existing footer (if the text deals with the addition of a single feature). Every article in the main tutorial line should contain the full footer as it should look at that point in the series. For this tutorial, our essential footer is:

```
Blender.Redraw()
```

3.39.3 Adding the Player Object

To start, create a new document and delete the basic cube. Create a Monkey/Suzanne. In the Editing panel with the monkey selected, change the mesh name (ME:) under Link and Materials to “Hero”. Click the F next to the input box to preserve the data block even when nothing links to it. Now delete the monkey object.

The following code will create an object called “Player” in the library and link it to the Hero mesh:

```
player = Blender.Object.New("Mesh", "Player")
player.link(bpy.data.meshes["Hero"])
```

In the first line, `Blender.Object.New` obviously references a new object. The “Mesh” variable should not be changed for the purposes of this code. I don't know exactly its function, so I don't want to give out misinformation, but I speculate that it has to do with the object type. If the object were to be a lamp or camera type, for example, you would not be able to apply a mesh to it. The second variable, “Player”, is the name of the actual object you're creating. Change it to your liking.

In the second line, we link `player` (which was equated to our new object) to the pre-existing mesh “Hero”, which is the Suzanne mesh we dealt with before. Using this method, you can model your character's mesh beforehand but have the actual character created dynamically. Using a complication of this code, you can make `player.link()` link to a variable and not the bit “`bpy.data.meshes[]`”. That variable can reference an existing mesh or it can create a new one. As for the meaning of the link's value, `bpy.data.meshes` is an array containing all the meshes in the movie. Likewise, `bpy.data.objects` contains all the objects in the movie. By going to the Scripts window and going to **Scripts->System->Interactive Console** you can gain such information as the contents of these arrays. By entering “`list(bpy.data.objects)`” into the console, you will be rewarded with a list in the format of `[Object "Camera"], [Object "Cube"], [Object "Lamp"]]` which is the list of objects in the standard new document set-up. So, to reference the item `[Object "Cube"]` you would use the line `“bpy.data.objects['Cube']”` and so on. For these commands to work, you must make sure to import `bpy` in your import section.

Appending the Object to the Scene

Like you must append the mesh to the object, you must also append the object to the scene, or it will just be a floating data block and not actually appear anywhere.

```
scene = Blender.Scene.GetCurrent() scene.link(player)
```

If you were to test this code by pressing **ALT-P** while the mouse is hovering over the Text Editor window, it would create an object named Player with the monkey

“Hero” mesh at the origin point of your scene. Make sure you included this tutorial’s import section before all of the other code and the essential footer after all of other code.

version 2.7:

```
import bpy myMesh = bpy.data.meshes["Hero"]
# reference existing mesh player =
bpy.data.objects.new("Mesh", myMesh) # create
new object player.name = "Player" # give it a name
bpy.context.scene.objects.link(player) # link to the scene
to show
```

Chapter 4

Unit 4 - Taking Off with Advanced Tutorials

4.1 Introduction

Python is a powerful, high-level, dynamic language. The version of Python used in Blender 2.67 is 3.3. If you are unfamiliar with Python, start with the [Python book](#). If you are familiar with older (2.x) versions of Python, [this page](#) summarizes what's new in 3.x.

If you are familiar with Python scripting in older versions of Blender, be aware that 2.5x/2.6x is completely different; the old Blender module is gone. Much of the functionality is now available through the bpy module, but don't expect an exact 1:1 correspondence.

4.1.1 First Steps In Blender Scripting

Open a new, default Blender document. If you haven't customized your settings, there will be a Timeline  window along the bottom; change this to a Python Console . Perhaps increase its height to let you see more lines at once.

To start with, let's find out what objects are present in the document. At the “>>>” prompt, type

```
bpy.data.objects
```

You should see the response

```
<bpy_collection[3], BlendDataObjects>
```

which isn't actually all that informative. In fact what you have here is an *iterator*; to see its contents, just do the usual Python thing and convert it to a list. Try entering this:

```
list(bpy.data.objects) #or bpy.data.objects[:]
```

This time the response should be

```
[bpy.data.objects["Camera"], bpy.data.objects["Cube"],  
 bpy.data.objects["Lamp"]]
```

which shows you how to refer to the three default objects you can see in the 3D View window.

Let's get a reference to the Cube object for more convenient access: type

```
Cube = bpy.data.objects["Cube"]
```

Now let's try querying the value of one of its attributes: type

```
Cube.delta_location
```

You should see the response

```
Vector((0.0, 0.0, 0.0))
```

The Vector type comes from the mathutils module provided by Blender. But unlike bpy, this is not automatically imported into the Python Console. So let's bring it in for subsequent use: type

```
import mathutils
```

OK, now let's try changing the location of the default Cube: type

```
Cube.delta_location += mathutils.Vector((1, 1, 1))
```

(Note the doubled parentheses: mathutils.Vector takes a *single* argument, which is a tuple of X, Y and Z coordinate values.)

Were you watching the 3D View when you pressed ENTER ? You should have seen the cube jump to a different position. To make it move again, press UPARROW to bring back the above command, so you can execute it again with ENTER . As soon as you do this, the cube will jump another step, like before. And that's it—you're scripting!

4.1.2 The bpy Module

The contents of the bpy module are divided into several submodules, among which are:

- bpy.data — This is where you find the contents of the current document.
- bpy.types — information about the types of the objects in bpy.data.
- bpy.ops — *operators* perform the actual functions of Blender; these can be attached to hotkeys, menu items and buttons. And of course they can be invoked from a script. When you write an addon script, it will typically define new operators. Every operator must have a unique name.
- bpy.context — contains settings like the current 3D mode, which objects are selected, and so on. Also accessible in the Console window via the global variable C.
- bpy.props — functions for defining *properties*. These allow your scripts to attach custom information to a scene, that for example the user can adjust through interface elements to control the behaviour of the scripts.

4.1.3 The mathutils Module

The module mathutils defines several important classes that are used heavily in the rest of the Blender API.

- Vector: the representation of 2D or 3D coordinates.
- Matrix: the most general way of representing any kind of linear transformation.
- Euler: a straightforward way of representing rotations as a set of *Euler angles*, which are rotation angles about the X, Y and Z axes. Prone to well-known pitfalls such as **gimbal lock**.
- Quaternion: on the face of it, a more mathematically abstruse way of representing rotations. But in fact this has many nice properties, like absence of gimbal lock, and smoother interpolation between two arbitrary rotations. The latter is particularly important in character animation.
- Color: a representation of RGB colours and conversion to/from HSV space (no alpha channel).

4.1.4 See Also

- Blender User's Manual section on [scripting](#)
- Blender 2.67 API Reference

- Blender site Python scripts [catalog](#)
- Blender Addon Tutorial by Campbell Barton
- Scripting forum on [blenderartists.org](#)

4.2 Anatomy Of An Addon

4.2.1 Introduction

An *addon* is a Python script that can extend the functionality of Blender in various ways. It can reside in a file within your Blender user preferences directory, or it can be stored in a text block within the Blender document. In the former case, the addon needs to be *enabled* in each Blender document where you want to use it, by ticking the checkbox in its entry in the Add-Ons list in the User Preferences window for that document. In the latter case, the script can be run by pressing ALT + P in the Text Editor window; or you can tick the “Register” checkbox for that script in the Text Editor, to have the addon be automatically enabled as soon as the document is loaded into Blender.

The main thing an addon script typically does is define one or more new *operators*. All the user-interface functions in Blender are performed by operators; these are invoked by being associated with menu items, buttons or hotkeys. Each operator is a subclass of the bpy.types.Operator class. The name you give your class in Python is immaterial outside the script where it is defined; it will be referred to from the rest of Blender via its *operator name*, which must be unique within the document. In the same way, your script can invoke other operators defined elsewhere via Blender API functions that take their operator name.

4.2.2 Text Blocks

A Blender document can contain *text blocks*, which are not the same as *text objects* in a 3D scene (though the former can be converted to the latter). Besides generating text objects, a text block can serve any purpose you like; for example, use it to pass workflow instructions to a colleague along with the document; display a copyright or help message in the initial layout that a user sees on opening the document; or hold a Python script that can be run by the user to perform some useful action related to the document.

4.2.3 Your First Operator

Open a new, empty Blender document. Bring up the Text Editor  in any convenient window; you will see an empty, grey rectangle. Before you can type in text, you

need to create a text block; do so by clicking the large button labeled “New” in the window header. As soon as you do this, you should see a popup menu appear listing all the text blocks in your document, with the current (only) entry named “Text”. You should also see a red insertion cursor appear at the top left, indicating that you can start typing.

Unlike the Console Window, nothing is automatically imported for you. So as in any other Python script, you need to mention every module you want to access.

You define your operator as a subclass of a subclass of the bpy.types.Operator. It must have a bl_idname class attribute which gives the operator name, while bl_label gives the user-visible name that appears in the sidebar menu. The former must have valid Python syntax for a name, including a single dot; the part to the left of the dot must be the name of one of the valid categories of operators, which you can find by typing

```
dir(bpy.ops)
```

into the Console Window.

Let’s define an operator that will add a new tetrahedron object to the document. The class definition should start something like this:

```
class MakeTetrahedron(bpy.types.Operator) : bl_idname = "mesh.make_tetrahedron" bl_label = "Add Tetrahedron"
```

It has to define an invoke method, which starts off like this:

```
def invoke(self, context, event) :
```

The invoke method performs the actual function of the operator; after it has finished, it must return a value which is a set of strings, telling Blender things like whether the operator is still executing in a modal state, or whether the operation has finished, and if so whether it was successful or not. To indicate successful completion: end the invoke method with this:

```
return {"FINISHED"}
```

Anyway, let us compute the coordinates of the vertices for the tetrahedron: with edges with a length of 1 Blender unit, suitable values are $(0, -\frac{1}{\sqrt{3}}, 0), (\frac{1}{2}, \frac{1}{2\sqrt{3}}, 0)$, $(-\frac{1}{2}, \frac{1}{2\sqrt{3}}, 0)$ and $(0, 0, \sqrt{\frac{2}{3}})$. Or in Python:

```
Vertices = [mathutils.Vector((0, -1 / math.sqrt(3), 0)),  
          mathutils.Vector((0.5, 1 / (2 * math.sqrt(3)), 0)),  
          mathutils.Vector((-0.5, 1 / (2 * math.sqrt(3)), 0)),  
          mathutils.Vector((0, 0, math.sqrt(2 / 3))), ]
```

Then we create the mesh datablock, giving it the name “Tetrahedron”:

```
NewMesh = bpy.data.meshes.new("Tetrahedron")
```

Fill it in with the above vertex definitions and associated faces:

```
NewMesh.from_pydata \ ( Vertices, [], [[0, 1, 2], [0, 1, 3], [1, 2, 3], [2, 0, 3]] )
```

The Mesh.from_pydata method is not currently well documented in the Blender API, but its first argument is an array of Vector defining the vertices, the second argument is an array of edge definitions, and the third argument is an array of face definitions. Each edge or face is defined as a list of vertex indices (2 elements in each list for an edge, 3 or 4 for a face), 0-based in usual Python style, being indices into the array of vertex definitions that you passed. Note that you pass **either the edge definitions or the face definitions, but not both**: the other one should be passed as the empty list. If this function is not used correctly, it will cause Blender to crash.

We also need to add the following, to tell Blender the mesh has changed and needs updating (as if it couldn’t figure it out itself):

```
NewMesh.update()
```

(Omitting this produces a minor quirk: right-clicking the newly-created tetrahedron does not highlight its outline in orange as with other objects, though the problem goes away if you enter edit mode on the object and exit again.)

Now to create the object datablock (I also give it the name “Tetrahedron”), and link it to the mesh:

```
NewObj = bpy.data.objects.new("Tetrahedron", NewMesh)
```

But the object will not appear to the user until it is linked into the scene:

```
context.scene.objects.link(NewObj)
```

To recap, here is the complete script:

```
import math import bpy import mathutils class MakeTetrahedron(bpy.types.Operator) : bl_idname = "mesh.make_tetrahedron" bl_label = "Add Tetrahedron" def invoke(self, context, event) : Vertices = [mathutils.Vector((0, -1 / math.sqrt(3), 0)),  
          mathutils.Vector((0.5, 1 / (2 * math.sqrt(3)), 0)),  
          mathutils.Vector((-0.5, 1 / (2 * math.sqrt(3)), 0)),  
          mathutils.Vector((0, 0, math.sqrt(2 / 3))), ] NewMesh = bpy.data.meshes.new("Tetrahedron") NewMesh.from_pydata \ ( Vertices, [], [[0, 1, 2], [0, 1, 3], [1, 2, 3], [2, 0, 3]] ) NewMesh.update() NewObj = bpy.data.objects.new("Tetrahedron", NewMesh) context.scene.objects.link(NewObj) return {"FINISHED"} #end invoke #end MakeTetrahedron bpy.utils.register_class(MakeTetrahedron)
```

Note the addition of the `register_class` call, to make Blender add your operator to its built-in collection.

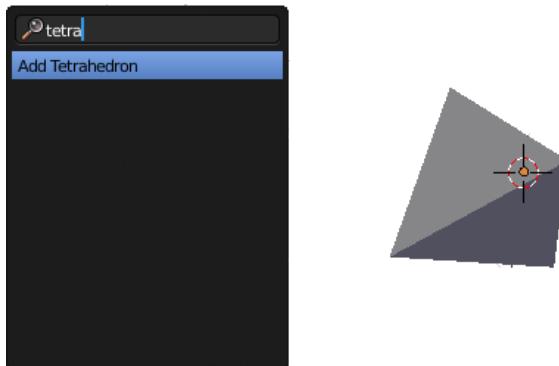
Now execute your script by typing ALT + P . If all goes well *nothing should appear to happen*; Blender will define the class and register it as a new operator as you requested, ready for use.

If you hit any syntax errors, Blender should display these in a popup window; go back and correct them, and re-execute the script with ALT + P .

4.2.4 Invoking Your Operator

We haven't (yet) defined any user interface for this operator; so how do we invoke it? Simple.

Go to a 3D View window. Delete the default cube to avoid it obscuring things, and press SPACE . This brings up a searchable menu of every operator defined for the current document. Into the search box, type part or all of the string you defined for the `bl_label` attribute above (typing "tetra" should probably be enough). This will restrict the menu to only showing items that contain that string, which should include the name of your operator; click with LMB on this or highlight it and press ENTER . If all goes well, you should see your tetrahedron object appear!

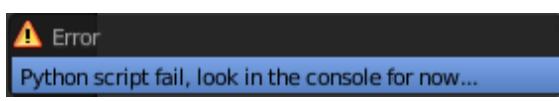


4.2.5 If You Hit An Error

If there is any error compiling or running the script, Blender should display this in a popup. For example, the following simple one-line script

```
raise RuntimeError("Uh-oh")
```

displays this popup:



The full Python traceback message is written to Standard Error, and looks something like this:

```
Traceback (most recent call last): File "/Text", line 1, in
<module> RuntimeError: Uh-Oh!
```

On Linux/Unix systems, the message will appear in the terminal session if you invoked Blender from the command line; otherwise it will be appended to your `~/.xsessionerrors` file if you launched Blender from a GUI. On Windows the message appears in the console window.

4.3 A User Interface For Your Addon

4.3.1 Start

So you've defined a new operator in your addon. Now it would be nice to give that addon a proper interface, so the user doesn't have to hunt through the spacebar menu to find that operator.

The nicest way to do this is to define a *panel* that shows up in a window somewhere, with controls in it that the user can click on to operate your addon. You can put the panel in all kinds of places, but here we will insert it in the Tool Shelf (which can be hidden or shown on the left side of the 3D View by pressing T .

4.3.2 Your First Panel

A panel is defined by subclassing the `bpy.types.Panel` class. You set the value of various attributes (`bl_space_type`, `bl_region_type`, `bl_category` and `bl_context`) to determine the context in which the panel will appear, and also give a title to the panel (`bl_label`):

```
class TetrahedronMakerPanel(bpy.types.Panel):
    bl_space_type = "VIEW_3D" bl_region_type = "TOOLS" bl_context = "objectmode" bl_category = "Create" bl_label = "Add Tetrahedron"
```

Those first three attributes cause the panel to appear in the Tool Shelf, but only while the 3D View is in Object mode. The `bl_category` line determines the toolbar tab the addon is placed in, and only applies to the toolbars with tabs (added 2.7). Specify any existing tab, or define a new one.

Your class also needs to define a `draw` method, which defines the items that go into the panel. This example creates a new column of UI elements for the panel, and inserts a single UI element which is a button that will invoke the operator with the name you previously defined when clicked:

```
def draw(self, context):
    self.layout.column(align=True)
```

```
TheCol.operator("mesh.make_tetrahedron", text="Add Tetrahedron") #end draw
```

4.3.3 Adding Undo Support

While we're at it, go back and add the following line to the operator definition:

```
bl_options = {"UNDO"}
```

This allows the user to undo the addition of the tetrahedron in the usual way with **CTRL + Z**, and redo it with **CTRL + SHIFT + Z**.

4.3.4 Put It All Together

Here is the complete script for the addon as it stands now:

```
import math import bpy import mathutils class TetrahedronMakerPanel(bpy.types.Panel):
    bl_space_type = "VIEW_3D" bl_region_type = "TOOLS"
    bl_context = "objectmode" bl_category = "Create" bl_label = "Add Tetrahedron" def draw(self, context):
    TheCol = self.layout.column(align=True)
    TheCol.operator("mesh.make_tetrahedron", text="Add Tetrahedron") #end draw #end TetrahedronMakerPanel
    class MakeTetrahedron(bpy.types.Operator):
        bl_idname = "mesh.make_tetrahedron" bl_label = "Add Tetrahedron" bl_options = {"UNDO"} def invoke(self, context, event):
            Vertices = \ [ mathutils.Vector((0, -1 / math.sqrt(3),0)), mathutils.Vector((0.5, 1 / (2 * math.sqrt(3)), 0)), mathutils.Vector((-0.5, 1 / (2 * math.sqrt(3)), 0)), mathutils.Vector((0, 0, math.sqrt(2 / 3))) ] NewMesh = bpy.data.meshes.new("Tetrahedron")
            NewMesh.from_pydata \ ( Vertices, [], [[0, 1, 2], [0, 1, 3], [1, 2, 3], [2, 0, 3]] ) NewMesh.update()
            NewObj = bpy.data.objects.new("Tetrahedron", NewMesh) context.scene.objects.link(NewObj) return {"FINISHED"} #end invoke #end MakeTetrahedron
    bpy.utils.register_class(MakeTetrahedron)
    bpy.utils.register_class(TetrahedronMakerPanel)
```

Note the addition of another `register_class` call for our custom panel.

As before, hit **ALT + P** to execute it. Nothing should appear to happen; Blender processes your subclass definitions, and registers them for use, as requested, in the appropriate places.

But now, go back to the 3D View. Make sure you are in Object mode. Bring up the Tool Shelf if it isn't visible, by pressing **T**. At the bottom; in the tab you have specified, you should see your new panel appear:

Note the triangle next to the title that Blender automatically gives you, to collapse or expand the panel without any extra work on your part. Get rid of any tetrahedron



object that might have been created by following the tutorial on the previous page; now click your new "Add Tetrahedron" button, and watch the object be created again!

4.3.5 Space Types, Region Types, Contexts, Oh My!

Those values for `bl_space_type`, `bl_region_type` and `bl_context` are (partially) listed [here](#) in the Blender API documentation, but not fully explained. Here's a list of permissible values for each that I found from looking at source code:

Presumably, not all combinations will make sense.

4.4 Adding A Custom Property

Addons will commonly do more than unconditionally perform a single action; the user may be able to control their actions in various ways. We can add controls for adjusting settings to our addon panel, and associate these with *custom properties* that we define for the current scene; Blender will then take care of updating the property values as the user operates those controls, and our operator's `invoke` routine can fetch those property values when it runs.

4.4.1 Defining A Property

Properties need to be defined at the same time our custom classes get registered. We previously did this in top-level statements, but now let's gather it all together into a `register` method, like this:

```
def register():
    bpy.utils.register_class(MakeTetrahedron)
    bpy.utils.register_class(TetrahedronMakerPanel)
    bpy.types.Scene.make_tetrahedron_inverted = bpy.props.BoolProperty \ ( name = "Upside Down",
    description = "Generate the tetrahedron upside down",
    default = False ) #end register
```

Here we are attaching the property as a new attribute of Blender's Scene class; Python lets us assign new attributes to just about any object, and Blender takes full advantage of that. Note that the property must be created using one of the property-definition routines provided in `bpy.props`:

choose the one that matches the type of property you want to create. Here we are defining a simple true/false toggle, which the user will control via a checkbox. Whatever name you use for your custom class attribute, an instance of that class will have an attribute with the same name, holding the actual value for that property.

The name will be used as the name of a control for examining or changing this property, while the description will appear as a tooltip when the user hovers the mouse over the control. The default is used as the initial value of the property.

Note also I tried to use a name, `make_tetrahedron_inverted`, which is less likely to clash with names defined by other addons or parts of Blender.

Let's also add an `unregister` method, which undoes everything that `register` does. This won't actually be used for now, but it will become relevant later when we extract the addon for separate installation:

```
def unregister() : bpy.utils.unregister_class(MakeTetrahedron)
bpy.utils.unregister_class(TetrahedronMakerPanel)
del bpy.types.Scene.make_tetrahedron_inverted #end unregister
```

To make the checkbox appear, add the following line to our panel's draw routine:

```
TheCol.prop(context.scene,
"make_tetrahedron_inverted")
```

The first argument to the `prop` method must be an instance of the class to which we attached our property definition above; in this case, it is the current scene.

Finally, we need to finish off our script with the following bit of boilerplate which will invoke our registration routine in the situations (like the Text Editor) where Blender doesn't do it for us:

```
if __name__ == "__main__" : register() #end if
```

4.4.2 Using The Property

Now we actually need to use the property in our operator's execute routine. We'll use it to negate a scale factor which will be applied to the Z coordinate of the vertices of the tetrahedron:

```
Scale = -1 if context.scene.make_tetrahedron_inverted
else 1
```

But only the last vertex has a nonzero Z coordinate, so this is the only one that needs a change in its computation:

```
Vertices = \[ mathutils.Vector((0, -1 / math.sqrt(3), 0)),
mathutils.Vector((0.5, 1 / (2 * math.sqrt(3)), 0)),
```

```
mathutils.Vector((-0.5, 1 / (2 * math.sqrt(3)), 0)),
mathutils.Vector((0, 0, Scale * math.sqrt(2 / 3))), ]
```

4.4.3 Put It All Together

To recap all our changes, here is the complete updated script:

```
import math import bpy import mathutils
class TetrahedronMakerPanel(bpy.types.Panel) :
    bl_space_type = "VIEW_3D" bl_region_type = "TOOLS"
    bl_context = "objectmode" bl_category = "Create"
    bl_label = "Add Tetrahedron" def draw(self, context) :
    TheCol = self.layout.column(alignment = True)
    TheCol.prop(context.scene, "make_tetrahedron_inverted")
TheCol.operator("mesh.make_tetrahedron", text = "Add Tetrahedron") #end draw #end TetrahedronMakerPanel
class MakeTetrahedron(bpy.types.Operator) :
    bl_idname = "mesh.make_tetrahedron" bl_label = "Add Tetrahedron" bl_options = {"UNDO"} def invoke(self, context, event) :
    Scale = -1 if context.scene.make_tetrahedron_inverted else 1
    Vertices = \[ mathutils.Vector((0, -1 / math.sqrt(3), 0)),
mathutils.Vector((0.5, 1 / (2 * math.sqrt(3)), 0)),
mathutils.Vector((-0.5, 1 / (2 * math.sqrt(3)), 0)),
mathutils.Vector((0, 0, Scale * math.sqrt(2 / 3))), ]
    NewMesh = bpy.data.meshes.new("Tetrahedron")
    NewMesh.from_pydata(\(Vertices, [], [[0, 1, 2], [0, 1, 3], [1, 2, 3], [2, 0, 3]])) NewMesh.update()
    NewObj = bpy.data.objects.new("Tetrahedron", NewMesh)
    context.scene.objects.link(NewObj) return {"FINISHED"} #end invoke #end MakeTetrahedron
def register() :
    bpy.utils.register_class(MakeTetrahedron)
    bpy.utils.register_class(TetrahedronMakerPanel)
bpy.types.Scene.make_tetrahedron_inverted = bpy.props.BoolProperty(name = "Upside Down", description = "Generate the tetrahedron upside down", default = False) #end register def unregister() :
    bpy.utils.unregister_class(MakeTetrahedron)
    bpy.utils.unregister_class(TetrahedronMakerPanel)
del bpy.types.Scene.make_tetrahedron_inverted #end unregister
if __name__ == "__main__" : register() #end if
```

As before, execute the script with ALT + P . Check the Tool Shelf in the 3D View, and your panel should now look like this:

Try executing it with and without the checkbox checked, and you should end up with two tetrahedra pointing in opposite directions.



4.5 A Separately Installable Addon

Embedding a script inside a Blender document is useful for some special purposes, but often you want to be able to reuse an addon across any number of Blender documents, and perhaps distribute it to others for use with their Blender projects. To do this, you will need to save the script as a text file that can be copied into your Blender user preferences. It will also need to have some additional information inserted.

4.5.1 The Addon Info Dictionary

In order for the addon to be installable by Blender, it will need to define a global called `bl_info`, the value of which is a dictionary. This contains various information about compatibility with versions of Blender, and also descriptive information to be shown to the user as they browse through the list of installable addons.

Here's what the info should look like:

```
bl_info = \ {
    "name": "Tetrahedron Maker",
    "author": "J Random Hacker <jrhacker@example.com>",
    "version": (1, 0, 0),
    "blender": (2, 5, 7),
    "location": "View 3D > Edit Mode > Tool Shelf",
    "description": "Generate a tetrahedron mesh",
    "warning": "",
    "wiki_url": "",
    "tracker_url": "",
    "category": "Add Mesh",
}
```

Most of the fields are informational. The “category” value must be one of the predefined ones that you will see listed in Blender’s User Preferences window when you switch to the “Add-Ons” tab; the addon display is sorted by category and by name within category, and the user can list scripts in all categories or in just one category.

The “version” field indicates the version of your script, and can be a tuple of any number of integers; the numbers are shown to the user joined with decimal points (e.g. “(1, 0, 0)” is displayed as “1.0.0”). The “blender” field indicates compatibility with versions of Blender; Blender versions older than this are supposed to reject the script. (Though what happens when a *newer* version introduces a backward-incompatible API change is not quite clear...)

4.5.2 Making The Addon Installable

The contents of your addon script are now complete. In the Text Editor, find the “Save As” option under the “Text” menu, and choose a filename for saving your script—perhaps call it `TetrahedronMaker.py`.

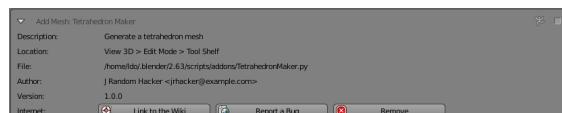
That's it. This saved text file is your complete addon, ready for installation into your Blender user settings, or distribution to others for installation in their settings.

4.5.3 Installing The Addon

To install the addon into your Blender user settings, go to the User Preferences, and bring up the Add-Ons tab. (It doesn't matter what document you might have open at this stage.) In the window header, click the “Install Add-On...” button, and find the `TetrahedronMaker.py` file you previously saved; select that, and click the “Install Add-On” button in the file selector. This copies the script file into your personal addons directory, which in Unix/Linux is `~/.blender/2.5x/scripts/addons/`. You can also directly copy your script file into this directory, instead of using the “Install Add-On...” button. To uninstall an addon, use the Remove button that appears in its info panel (see below) or delete the script from this directory, together with its `.pyc` (compiled) version (which you will find in the `__pycache__` subdirectory).

4.5.4 Using The Installed Addon

To make use of the installed addon, open a new Blender document. Go to the User Preferences window, and bring up the Add-Ons tab. Browse through the list of addons (feel free to restrict the display to the “Add Mesh” category as specified above) until you find your Tetrahedron Maker. You can click the white triangle to the left of the name to expand the list item display to show all the details:



Note the buttons that may appear along the bottom: the “Remove” button appears for your own user-installed scripts, not for ones installed systemwide; the “Link to the Wiki” button is only visible if you provided a URL in the “wiki_url” field of your `bl_info`; and “Report a Bug” comes up only if there is something in the “tracker_url” field.

Click the checkbox in the top right corner of the item display to enable the addon for this document.

Now switch to the 3D View, bring up the Tool Shelf if it's not already visible, and you should see that familiar panel

appear:



You know what to do next...

4.6 Object, Action, Settings

4.6.1 Prologue

Now that we've covered the basics of writing a working addon and making it installable, let's add some refinements to its functionality.

To start with, users normally add new objects via the Add menu which pops up when pressing SHIFT + A , rather than via a custom panel. Can our script add a new "Tetrahedron" item to that menu? Yes it can.

Also, our script currently requires us to adjust its setting (the "Upside-Down" checkbox) *before* it performs its action. In user-interface parlance, this is the ordering "Object→Setting→Action": select an object, specify the settings for the action, then perform the action. Whereas the preferred order in Blender 2.5x is "Object→Action→Setting": which means the action is performed with some initial settings, which the user is then free to modify while observing their effect. This gives a much smoother workflow, rather than the user having to guess what the effects will be before applying them, and then undoing and trying again if they guessed wrong.

(Of course, since our example script creates a new object rather than modifying an existing one, the initial "Object" step is not relevant here. But it is to other operators.)

So the modifications we need to make to our script are:

- Get rid of the existing Panel subclass.
- Our register function will add a new entry to the Add menu (specifically, the mesh-add submenu) to invoke our Operator subclass.
- When our operator is invoked, a panel will appear, with all the settings controls we previously had, except for the "Add" button. The user can play with these settings, and observe the effect on the newly-created object immediately.

The nice thing is, Blender does most of the hard work for us, so we only need to edit a few lines of code to achieve all the above!

4.6.2 Adding To The Add Menu

This needs to be done in two steps. First we need a function which will be invoked when Blender creates its Add menu: this will add an entry that invokes our custom operator by its name. Specifically, we will add our custom item to the "Mesh" submenu of the "Add" menu. We can also assign an icon to the menu item; here I'm using the generic plugin icon:

```
def add_to_menu(self, context):
    self.layout.operator("mesh.make_tetrahedron", icon = "PLUGIN") #end add_to_menu
# version 2.72b only allows for the icon value from the set:
# ("NONE", "QUESTION", "ERROR", "CANCEL",
# "TRIA_RIGHT", "TRIA_DOWN", # "TRIA_LEFT",
# "TRIA_UP", "ARROW_LEFTRIGHT", "PLUS",
# "DISCLOSURE_TRI_DOWN", # "DISCLOSURE_TRI_RIGHT", "RADIOBUT_OFF", "RADIOBUT_ON", "MENU")
```

Having done that, we change our register function to add our add_to_menu item to a list that Blender uses to create its Add→Mesh menu:

```
def register():
    bpy.utils.register_class(MakeTetrahedron)
    bpy.types.INFO_MT_mesh_add.append(add_to_menu)
#end register
```

Note this bpy.types.INFO_MT_mesh_add object is not currently mentioned in the API documentation; I found it by examining the scripts that come bundled with Blender.

And of course we should clean up ourselves, so the unregister function needs to remove the item we added:

```
def unregister():
    bpy.utils.unregister_class(MakeTetrahedron)
    bpy.types.INFO_MT_mesh_add.remove(add_to_menu)
#end unregister
```

4.6.3 Fixing Up The UI

Note that previously, our register and unregister functions were attaching a custom property to Blender's Scene class to hold our setting value. That code is now gone, but we still need the property. Instead, it will now be attached directly to our MakeTetrahedron class.

Also, remember we said we were getting rid of the TetrahedronMakerPanel class? In fact we keep the draw method from that, and move it to our MakeTetrahedron operator, only getting rid of the last button in the panel for invoking the operator. This is because by the time the panel is being drawn, the operator has already been invoked.

We also need to tell Blender that we are conforming to the Object→Action→Settings convention, by adding the "REGISTER" item to our bl_options.

So the header of our operator class now looks like this:

```
class MakeTetrahedron(bpy.types.Operator) : bl_idname = "mesh.make_tetrahedron" bl_label = "Tetrahedron" bl_options = {"REGISTER", "UNDO"} inverted = bpy.props.BoolProperty \ ( name = "Upside Down", description = "Generate the tetrahedron upside down", default = False ) def draw(self, context) : TheCol = self.layout.column(align = True) TheCol.prop(self, "inverted") #end draw
```

Previously our custom property was called `make_tetrahedron_inverted`, but here I just call it `inverted`, because after all it is being attached to our own class, so there should be no worry about name clashes.

Note that I also changed the `bl_label` text, taking out the word “Add”. This is redundant, because our label will be appearing in a menu that is already titled “Add”.

4.6.4 invoke Versus execute

The last thing to do is rearrange the code for actually creating the tetrahedron object. Previously we had it in a method called `invoke`, and that method will still be called when our operator is selected from the Add→Mesh menu. Then our `draw` method will be called, so our panel will also appear. But then, if the user makes adjustments to the controls in our panel, Blender will invoke a different operator method, called `execute`.

To the user, it looks like they are making adjustments to an already-created object. But in fact our `execute` method can do exactly the same thing as `invoke`, namely create a new object every time it’s called! Blender will take care of getting rid of previously-created objects, so the user won’t know the difference.

Nice, isn’t it? But in order for this to work, our code needs to do one thing more: ensure that our newly-created object is the only selected, active object. So the following is added after `NewObj` has been created and linked into the scene:

```
bpy.ops.object.select_all(action = "DESELECT")  
NewObj.select = True context.scene.objects.active = NewObj
```

4.6.5 Put It All Together

Note that the object-creation code has been moved into a common routine that I have called `action_common`. This can then be called from both the `invoke` and `execute` methods.

```
import math import bpy import mathutils class MakeTetrahedron(bpy.types.Operator) : bl_idname = "mesh.make_tetrahedron" bl_label = "Tetrahedron" bl_options = {"REGISTER", "UNDO"} inverted =
```

```
bpy.props.BoolProperty \ ( name = "Upside Down", description = "Generate the tetrahedron upside down", default = False ) def draw(self, context) : TheCol = self.layout.column(align = True) TheCol.prop(self, "inverted") #end draw def action_common(self, context) : Scale = -1 if self.inverted else 1 Vertices = \ [ mathutils.Vector((0, -1 / math.sqrt(3), 0)), mathutils.Vector((0.5, 1 / (2 * math.sqrt(3)), 0)), mathutils.Vector((-0.5, 1 / (2 * math.sqrt(3)), 0)), mathutils.Vector((0, 0, Scale * math.sqrt(2 / 3))) ] NewMesh = bpy.data.meshes.new("Tetrahedron") NewMesh.from_pydata \ ( Vertices, [], [[0, 1, 2], [0, 1, 3], [1, 2, 3], [2, 0, 3]] ) NewMesh.update() NewObj = bpy.data.objects.new("Tetrahedron", NewMesh) context.scene.objects.link(NewObj) bpy.ops.object.select_all(action = "DESELECT") NewObj.select = True context.scene.objects.active = NewObj #end action_common def execute(self, context) : self.action_common(context) return {"FINISHED"} #end execute def invoke(self, context, event) : self.action_common(context) return {"FINISHED"} #end invoke #end MakeTetrahedron def add_to_menu(self, context) : self.layout.operator("mesh.make_tetrahedron", icon = "PLUGIN") #end add_to_menu def register() : bpy.utils.register_class(MakeTetrahedron) bpy.types.INFO_MT_mesh_add.append(add_to_menu) #end register def unregister() : bpy.utils.unregister_class(MakeTetrahedron) bpy.types.INFO_MT_mesh_add.remove(add_to_menu) #end unregister if __name__ == "__main__" : register() #end if
```

4.6.6 Undocumented Blender

That `INFO_MT_mesh_add` object is one of a bunch of undocumented things lurking in the `bpy.types` module. They all have names of the form `prefix_type_restofname`, where `prefix` is an all-uppercase mnemonic for the category of object (mostly a window type, e.g. `INFO` for the Info window, which is where the main menu bar appears, though there are also names for major object categories like `MESH` and `LAMP`), and `type` indicates the class of object: `HT` for a window header UI object (`bpy.types.Header`), `MT` for a menu (`bpy.types.Menu`), `OT` for an operator, and `PT` for a panel (`bpy.types.Panel`). The `OT` ones seem to correspond directly to objects in `bpy.ops`, but at a lower level; it’s probably easier just to use the official `bpy.ops` objects. As for the `HT`, `MT` and `PT` objects, these all have `append`, `prepend` and `remove` methods that you can use to customize them: `append` adds a widget at the end (right or bottom), while `prepend` puts it at the front (left or top).

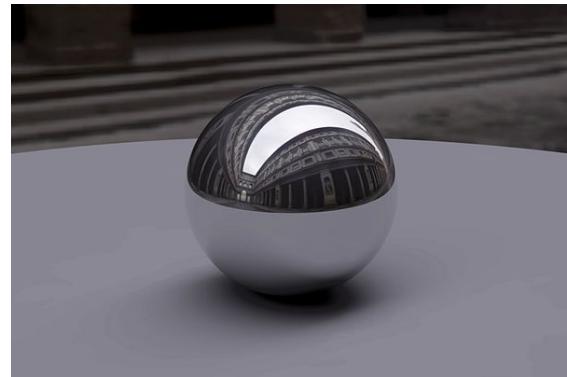
Since they aren’t (yet) mentioned in the official Blender documentation, here is a list of all the ones I’ve been able to find:

4.7 Overview

4.7.1 Advanced Modeling

We have learned quite a few useful techniques that we can now use to make something really cool or fun. But there are plenty of little tricks hidden within Blender that will allow us to make very realistic or intricate 3D graphics with a much lower amount of memory used. It usually takes an extra application or two, and the more advanced the program, the better; however, in the next few tutorials the *Paint* program--or its equivalent on another OS--will be just fine for our purposes.

The purpose of these modeling tutorials will be to lower the poly count on your models--this could range from no effect to cutting out 99.99% of your original faces, and still being able to keep the realistic look you were going for. Make sure you have read over the other tutorials in this book or at least are very familiar with Blender, so you can better understand how to work these modeling techniques.



HDR image file example

HDRI stands for **High Dynamic Range Imaging**, and is basically an image format that contains from the deepest shadow up to the brightest highlight information. While an 'ordinary' digital image contains only 8 bits of information per color (red, green, blue) which gives you 256 gradations per color, the HDR image format stores the 3 colors with floating point accuracy. Thus the 'depth' from dark to light per color is virtually unlimited. Using HDR images in a 3D environment will result in very realistic and convincing shadows, highlights and reflections. This is very important for realistic emulation of chrome for example.

4.8 High Dynamic Range imaging (HDRI)

4.8.1 Introduction

You may have heard various people talk about HDR images. (WETA, Lucas, even Tim Sweeny). HDR images are part of a technology called HDRI which stands for "High Dynamic Range (image)". So... what on earth does that mean?

Here's a link to Wikipedia's article on the [HDR format](#) which I personally give all my credit to Paul Debevec for putting it to use for computer graphics purposes. Any-way, before you start trying to understand the usefulness of HDRI, please read the wikipedia link.

Also, visit [Paul Debevec's website](#) if you've got some more time to spare.

To sum up the excitement of HDR CG, think of it like the hype of the next-generation videogames that are about to come out, except set the stage for 1996 instead of 2006. Paul Debevec pioneered parallax mapping, HDR lighting, image-based modeling, his latest work includes some even more amazing technologies, and for the record he's my hero too.

To use HDRI images for 3D rendering, you need something called a **light probe**...

4.8.2 Definitions

HDRI

Light Probe

A Light Probe is a HDR image containing 360 by 360 degrees ([solid angle \$4\pi\$ steradians](#)) image information. In other words : it's a 360 degree spherical panorama image, not only looking around the horizon, but also up and down. Thus a Light Probe image contains all visible information as can be seen from a specific point, wherever you turn your head.

4.8.3 Usage

Given that a Light Probe image is an 'all around' image with a high dynamic range, it's the perfect solution for your 'world' background, especially for a 3D animation.

4.8.4 Quick Tutorial (*for experienced blendies*)

First of all, you'll need an HDR image. There is a whole range at <http://debevec.org/Probes/> that you can download for free. (There are even more at <http://blenderartists.org/forum/showthread.php?t=24038>). I will use the St. Peter's Basilica probe, but any other HDR image will do just fine.

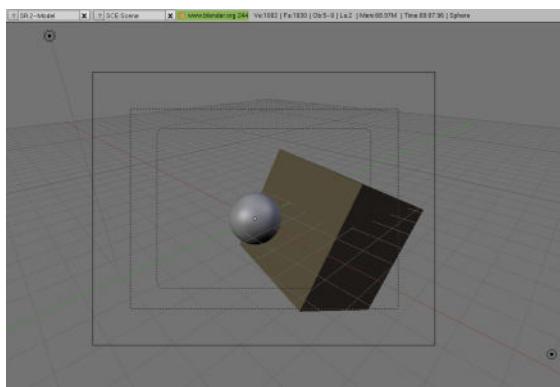
To apply the HDRI environment to your scene:

- Go to the shading settings (press F5) and click the World button.
- Enable “Real” to force the horizon to stay still, as opposed to follow the camera
- If you’re rendering with Blender Internal (ver. 2.49), go to “Ambient Occlusion” tab and activate “Ambient Occlusion”, then activate “Sky Texture”. Skip this step if you’re using external renderers.
- In the “Texture and Input” tab, click “Add New” and “Angmap”.
- Then go to the “Map To” tab and deactivate “Blend” and activate “Hori”.
- Now go to the Texture settings (press F6) and change the “Texture Type” to “Image”.
- Click the “Load Image” button and locate your HDR image.
- If you’re using YafRay or other external renderers, you need to turn on Global Illumination and to set the quality to something other than “none”.

4.8.5 Step-by-step Tutorial

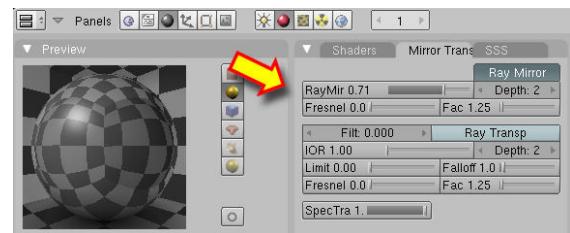
Build a simple scene

To see the advantage of using a 360 by 360 world image, the simplest example to demonstrate this is a scene with a mirrored sphere.



Scene

1. Add a sphere and a cube to your scene and place them in a bit of an interesting position. (note that I added a second lamp to light up the shadow part of the cube)
2. Perhaps give the cube a different color than the default grey.

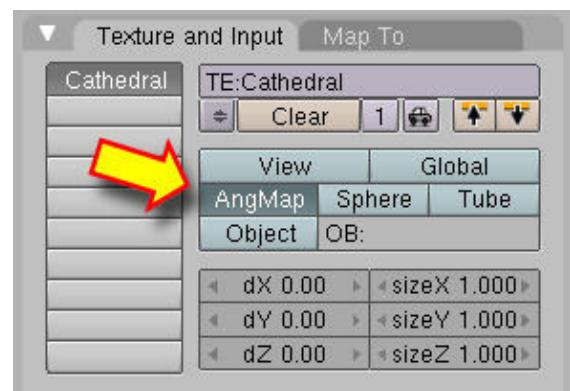


Material setting for a mirror surface and a preview of it

1. Give the sphere a mirrored material : go to the **Shading -> Material** panel (F5) and find the **Mirror Trans** buttons.
2. Check that **Ray Mirror** button is pressed. If it is not, check it.
3. Set the **RayMir** value to a value of 0.5 or higher. Your preview should show the reflection of the checkboard environment.

Render with HDR (Blender Internal v2.49)

1. Download a HDR image (see Paul Debevec’s website)
2. Go to the shading settings (press F5) and click the World button.



AngMap enabled

1. Go to “Ambient Occlusion” tab and activate “Ambient Occlusion”, then make sure “Raytrace” is chosen as gather method, and activate “Sky Texture”
2. Adjust quality settings:
 - Increasing number of samples reduces noise, but increases render times
 - Adaptive QMC is faster, but generates more noise than Constant QMC
3. In the “Texture and Input” tab, click “Add New” and “Angmap”

4. Then go to the “Map To” tab and deactivate “Blend” and activate “Hori”.
5. Now go to the Texture settings (press F6) and change the “Texture Type” to “Image”.
6. Click the “Load Image” button and locate your HDR image.

Render with HDR (Blender Internal v2.69)

1. Download a HDR image (see Paul Debevec’s website)
2. Go to Properties tab and click the World button. In the World Section check the “Real Sky” Button
3. (optional) Go to “Ambient Occlusion” section and activate “Ambient Occlusion”, then make sure “Raytrace” is chosen as gather method in Gather section
4. (optional) Adjust quality settings:
 - (a) Increasing number of samples reduces noise, but increases render times
 - (b) Adaptive QMC is faster, but generates more noise than Constant QMC
5. In the Properties tab click Texture button->then click on 'show world texture'->then select type as 'image or movie'->then in mapping section select coordinates as 'Angmap'->then in influence section uncheck the 'blend' button and check the 'horizon' button.
6. In the Image section click on open button and locate your HDR image and Load it.

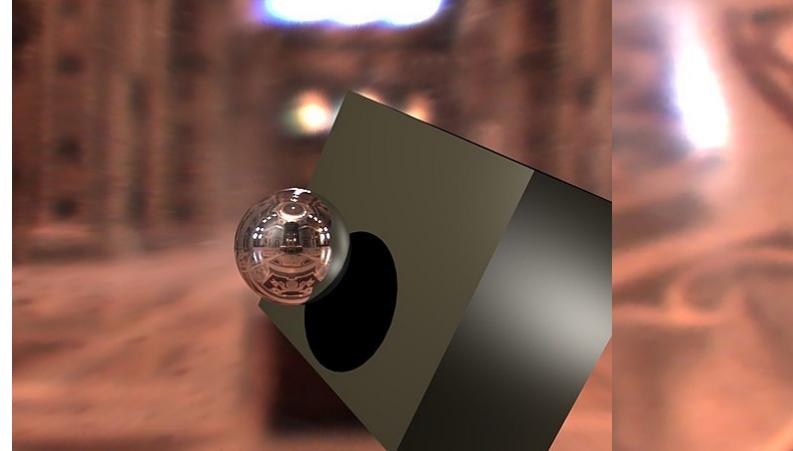
Render with YafRay

1. Download a HDR image (see Paul Debevec’s website).
2. Go to the shading settings (press F5) and click the World button.
3. In the “Texture and Input” tab, click “Add New” and “Angmap”. Note that the Angmap button is the important thing to tell Blender that this file is a Light Probe file !
4. Then go to the “Map To” tab and deactivate “Blend” and activate “Hori”.
5. Now go to the Texture settings (press F6) and change the “Texture Type” to “Image”.
6. Click the “Load Image” button and locate your HDR image.

(optional step, as it was not needed for my setup :)

5. Press F10 and change the “Blender Internal” to “YafRay”. You need to turn on Global Illumination and to set the quality to something other than “none”. Note that the YafRay renderer does not come standard with the Blender installation. You need to download and install this separately.

Your result should look like this: (Rendering: left with Blender, right with YafRay) Click for larger version



Note that the reflecting ball reflects the whole interior from every angle, even though we added just a single image to the World settings !

4.9 Creating a Light Probe

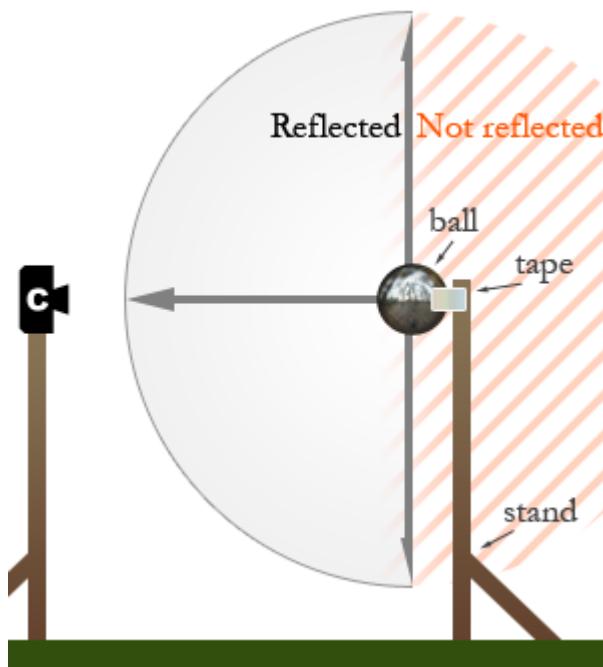
The light probe is, in the simplest terms, a photograph of your environment. they work in very much the same way that reflection maps do, and are made the same way.

Equipment:

1. A pure silver ball. Try a plastic Christmas tree ball ornament.
2. A camera, preferably digital. If you have a high-end digital camera, you'll have less work ahead.
3. A place you'd like to capture the lighting from. Try laying different things in your environment to get a good idea.
4. Something to fire the shutter without touching the camera. For digital cameras, Paul Debevec recommends using a program that will take all the pictures for you.
5. A tripod.

Set it up like so. Remember that the height of your camera and the height of the ball relative to each other controls the angle at which the horizon will be shot. In other words, shoot the ball at the same angle that you plan to

shoot your 3D scene in. If your scene is animated... consider making a similar rig except attaching your reflective ball to a video camera as was done for *Flight of the Navigator*.



(note from a VFX pro who is using this technique for years: on a mirror ball everything is reflected except the area right behind the sphere opposite to the camera - where the tape is on the above picture)

The process:

1. Set up your rig and take a series of images with varying exposure times.
2. Taking 2 sets of pictures, each offset by 90 degrees, will enable you to get better coverage of the background and eliminate the reflection of the camera taking the picture.

4.10 Landscape Modeling with Heightmaps

This tutorial will show you how to make advanced terrain such as mountains using Blender and gimp or any other image editing software. Blender has the ability to use height maps to create meshes. Height maps are black and white images with white representing the highest point and black the lowest.

4.10.1 Creating the heightmap image

Note: This entire Tutorial presumes that you are already familiar with other Editing Software, such as GIMP...

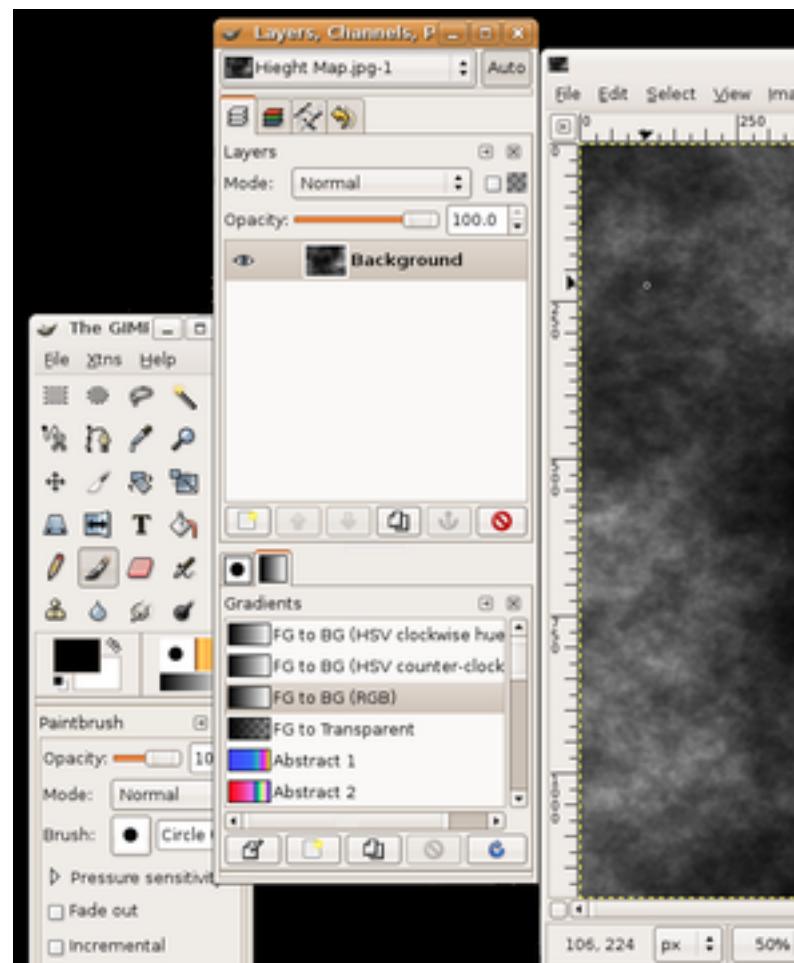
and already know how to create “textures” with that software. However, you do not have to use another program to obtain a texture. At the end of the next section, they show you how to generate a random ‘cloud’ texture which you can use directly. Do not worry about GIMP or Photoshop. Just skip the rest of this section and most of the next one.

To begin with, open your image editing software. This part applies to Gimp only (downloadable at <http://gimp.org>), if you use another program you will have to do it another way.

First use the “New” menu option to create an image 1600 wide by 1200 tall. Go to **Filters** → *Render* → *Clouds* → *Plasma* (In Photoshop, this is Filter > Render > Clouds). For this example just use the default settings, it doesn't really matter. Click OK. You should now have a nice colorful image. We don't want that, we want it in black and white. Make sure your Gradient is set to “FG to BG(RGB)”(this is the default anyway) and that your foreground color is black and the background color is white. Go to **Colors** → *Map* → *Gradient Map*. This will convert it to black and white for you.

Save your image as PNG or JPEG.

It should look something like this



4.10.2 Create grid and add the image as texture

Open Blender and delete the default cube.

Add a grid (**SPACE → Add → Mesh → Grid**) with resolutions 32 and 32 from top view. Do not scale the grid just yet, zoom instead if you wish to take a closer look at your grid. It will be explained later why you shouldn't scale now.

- **Response 1:** You should create the basic shape and outline in object mode, then edit the details in edit mode, kinda like the names imply.
- **Response 2:** When you create an object using the Front or Side views while having **Aligned to View** selected in the **Edit Methods** in **Blender Settings Panel**, the Local Z axis of the object will be pointing to the Global Y or X axis, respectively. For the matter of this tutorial, if you do that, you'll be forced to rotate the object 90° in X or Y, before going on. That's where problems may begin, if you rotate it in **Edit Mode** which doesn't change local axis orientations. In that case you'll be forced to reset transformations (**Ctrl+A -> Scale and Rotation to ObData**). However, if you rotate it in **Object Mode**, and if you make sure the Local Z axis points upwards after the rotation, you should be seeing good results later on. (*To see the Local orientations of an object, Press F7 for the Object Panel (the one with three arrows, 4th from the left) and in the Draw pane click the Axis button.*)

Click **F5** to go to the *Shading panel*. By default the *Material buttons* button should be selected (button with red ball), but if it isn't then click on it. If this window has a panel named *Preview* and it is empty then look to the right for the *Links and Pipeline* panel. Here you should find an *Add New* button beneath *Link to Object*, click on it. 3 more panels should appear when you do this. You should find the *Texture buttons* button, it is just to the right of the *Material button* and is black & yellow. You may also just hit **F6**. Now you should get two new panels titled *Preview* and *Texture*, in the *Texture* panel you'll find an *Add new* button, click it. Where the button just was it should now be a text field saying something like "TE:Tex.001", click in the text field and write "height" instead. Look a bit to the right and down for a dropdown menu allowing you to choose *Texture Type*, click it and select *Image*. Two more panels will appear and the rightmost should be *Image* with a *Load* button.

Before you load the image take a look at the *Map Im-*

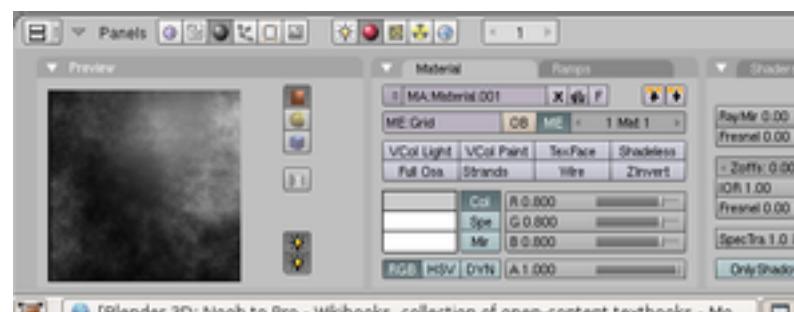
age panel. About in the middle there should be 5 buttons, *Extend*, *Clip*, *ClipCube*, *Repeat* and *Checker*. Select *Clip* or your heightmap may wrap around the grid (if this doesn't make sense then do this tutorial twice, one where you keep *Repeat* selected and one without. In that case make sure there's some bright white spots in your height map on each edge).

Click the *Load* button and load the heightmap you just created in your favourite image editing software.

If you're going for the random cloud-type image, though, you can make it from within Blender itself - when you add a texture, in the menu where you select 'image,' before actually selecting it, pick 'clouds' instead, and play with the settings.



Click the *Material buttons* button again (the red ball button) and look in the *Preview* panel. If this just shows a black ball then click on the uppermost button just to the right of the black ball.



4.10.3 Use texture as heightmap

(Blender 2.5+ note: Now you do this using a "Displace" modifier that applies the Texture you created.)

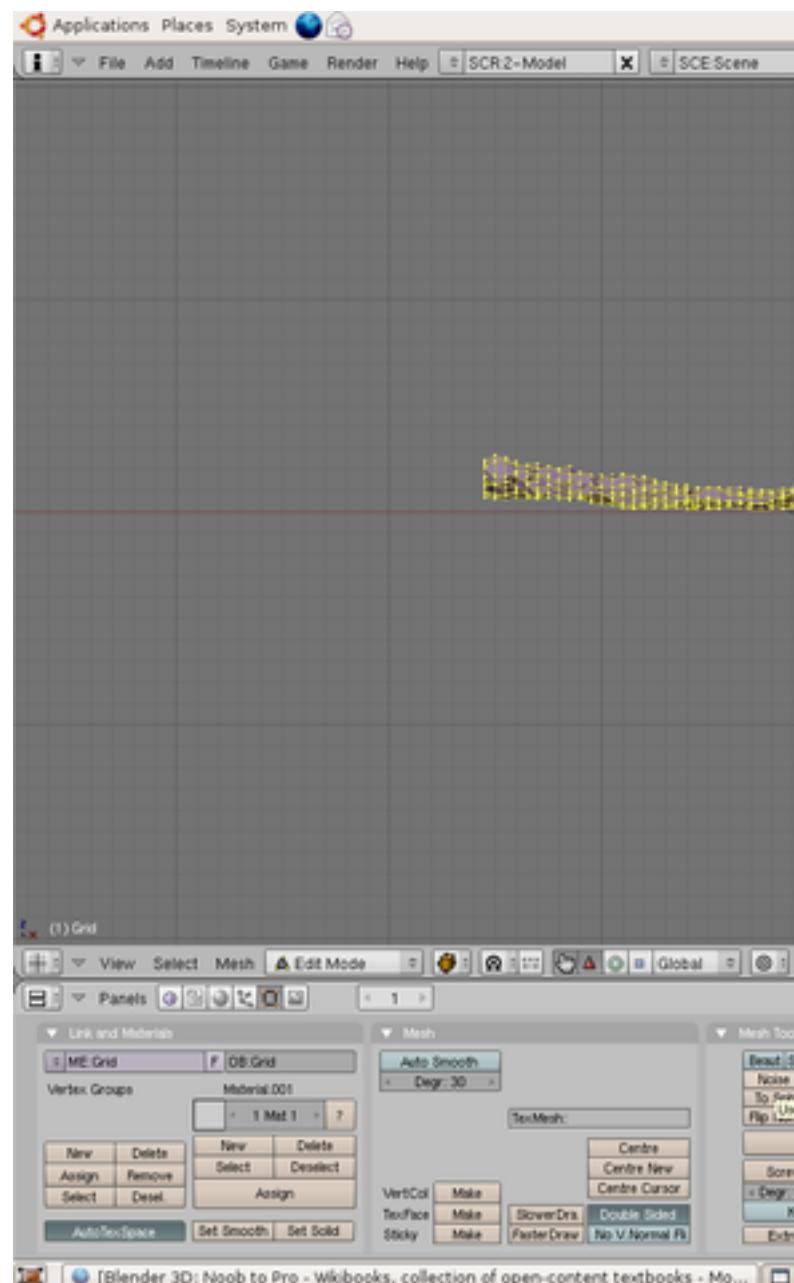
Look to the top right of the rightmost panel, the window should have 3 panes titled *Texture*, *Map Input* and *Map To*. Click on the *Map To* pane. A whole lot of buttons will appear:

- Turn off the *Col* button. "Col" stands for "color," and we don't want our plane to be *colored* by this texture.
- All of these buttons should be turned off: we're not going to be using this texture to "Map To" anything.

Instead, we'll be using it to deform the actual geometry.

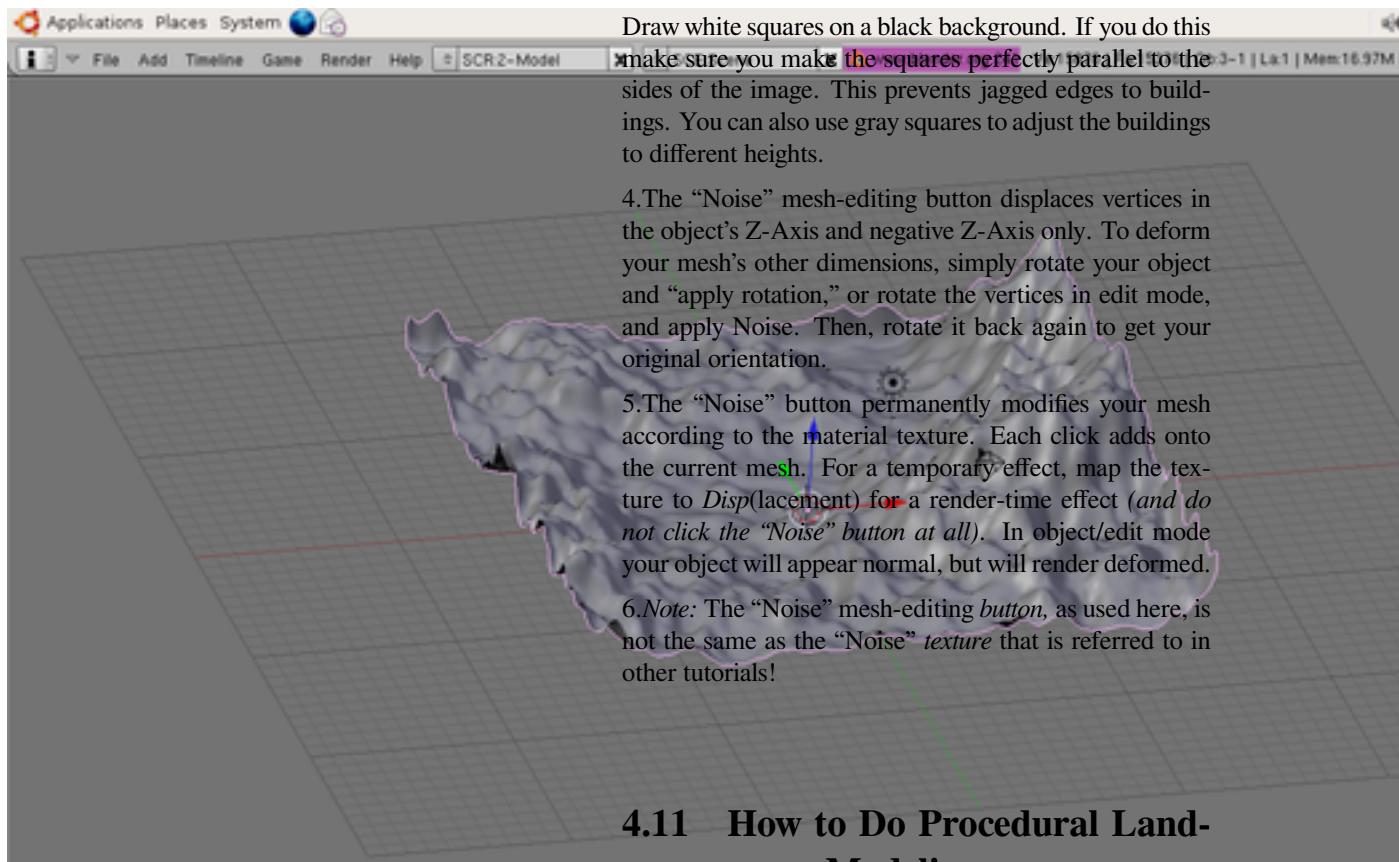
Go to the *Edit window* (**F9**) and enter edit mode (**TAB**) if you're in *Object Mode*. Go to either front or side view so you can see what you are doing. Click the *Noise* button in the *Mesh Tools*-panel (it's located in the top left corner of this panel) and your grid should start to change shape.

Note: make sure all your vertices are selected with AKEY or whatever you want to use to select. If not selected, *Noise* won't do anything!

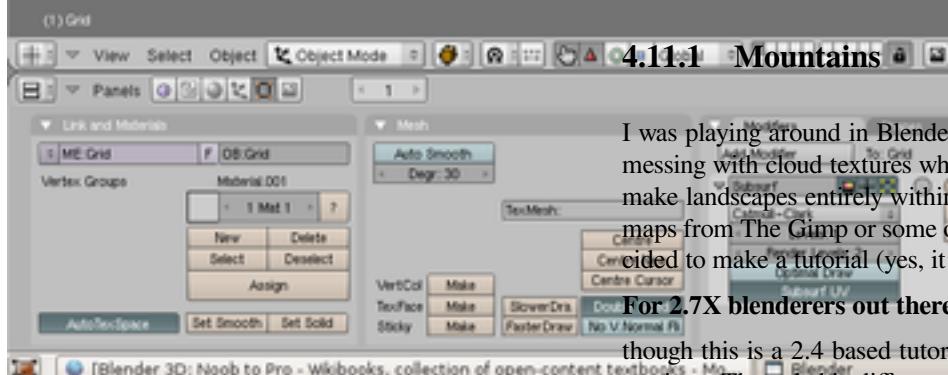


Continue clicking the button until the terrain has reached the height you want and voilà, you have just made terrain using heightmaps. You may want to subsurf the shape to get a smoother effect. If you wish to scale your new landscape then now is the time. There seems to be a bug or gotcha causing your heightmap to be tiled on your grid rather than resized if you scale the grid before applying the heightmap. Hopefully someone familiar with blender will clarify how this should be done properly, but until then this method will work.

- (While you could, theoretically, use this to your advantage ... intentionally de-selecting vertices that you don't want to influence ... in practice that's quite awkward if for some reason you have to do it again. It's much easier to leave any such areas, that you don't want to influence, "completely white" in the image.)



4.11 How to Do Procedural Landscape Modeling



Tips:

1.If you want a more jagged landscape then adjust the contrast on the image editing software

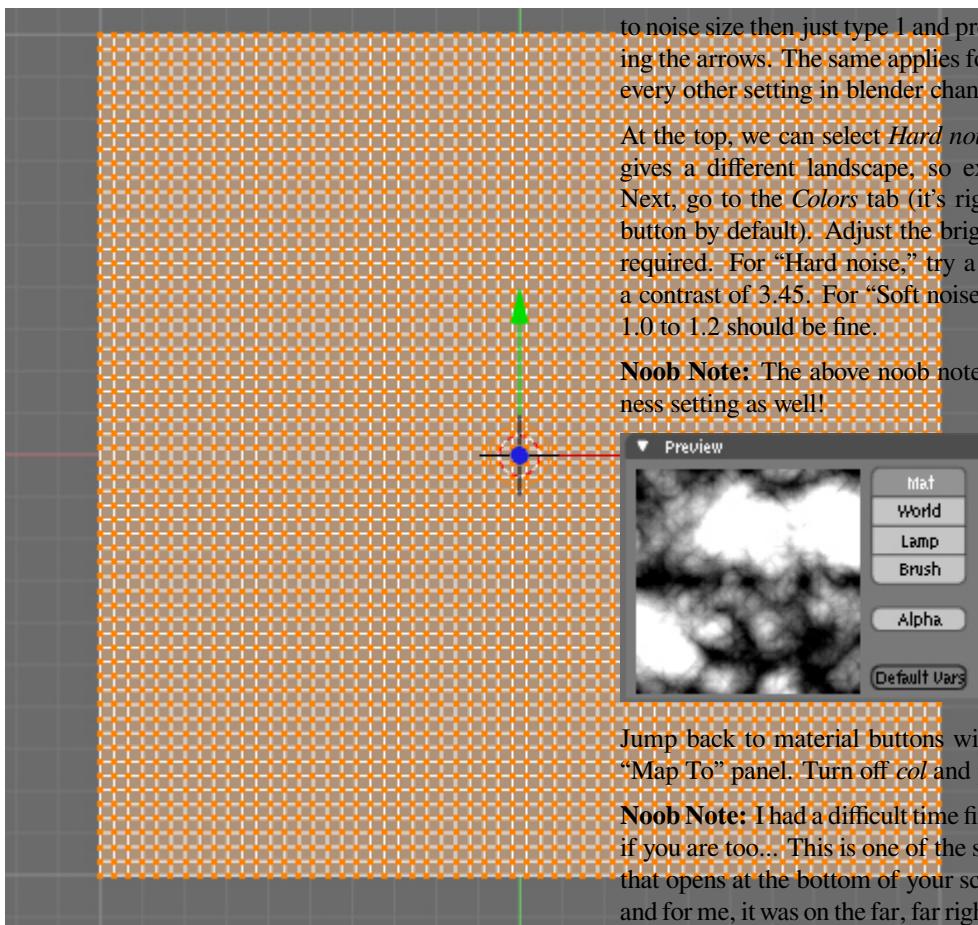
2.Once you've finished using the *Noise* button in the *Mesh Tools* panel as shown above, the texture is no longer needed: the effect it has made on the geometry of the object is permanent. If texture-resources are tight, you can remove it now. But it's advisable to keep it, unchanged, in case you need to re-do your work at some time in the future. If you want to use other kinds of texture-maps (*Color*, *Normal* maps, and so-forth), you can use the landscape-image as a handy reference (say in a background-layer in your painting program) for correct placement of these features, but you should not alter the landscape-image itself.

3.Height maps can also be used to make city terrain.

Making the Mesh

To start, open up Blender and with the cube selected, press the **X**KEY and click Erase Selected Objects.

Add a grid (**SPACE--> Add--> Mesh--> Grid**), using whatever numbers you like, remembering that bigger numbers means more vertices. For this tutorial, I'll use 60 for x and y resolution. Scale the grid up, but only a little bit (I went up to 1.3) as the texture won't make very tall landscape with a very big grid. Press the **F5** key to go to the Materials tab, then add a new material.



Displace and Shade

Now press the **F6** key to go the Textures panel, and add a cloud texture in the very first slot (slot 0).

After Pressing F6, look at the windows on the bottom. To the left is the “preview” window and next to that (to the right) is a *Texture* window with an *Add* button... click that and then go to the little pull down menu that shows up just below and to the right. From there, find and select “Clouds”... and now, continue!

You may be wondering how we are going to make landscape with a cloud texture, when you could just use the Gimp height map plugin, Terragen, or the A.N.T. modeling script right in Blender. Well, we are going to make the landscape with a **height map**. The color determines the height: white = very tall; grey = halfway tall; black = no height. This makes the cloud texture a very good candidate for land.

The next thing we are going to do is adjust the size. You can set the noise size to whatever you want, but I want to have several bigger pieces of landscape, instead of a bunch of little ones so I'm setting noise size to 1.00 and noise depth to 6, as I want a high level of detail. (The *Preview* immediately shows you the effect of your “tweaks” as you make them.)

Noob Note: You can hold shift and click the number next

to noise size then just type 1 and press enter instead of using the arrows. The same applies for NoiseDepth, almost every other setting in blender changes like this as well.

At the top, we can select *Hard noise* or *Soft noise*. Each gives a different landscape, so experiment with them. Next, go to the *Colors* tab (it's right next to the texture button by default). Adjust the brightness and contrast as required. For “Hard noise,” try a brightness of 1.9 and a contrast of 3.45. For “Soft noise,” a brightness around 1.0 to 1.2 should be fine.

Noob Note: The above noob note applies to the brightness setting as well!



Jump back to material buttons with **F5** and go into the “Map To” panel. Turn off *col* and press *nor* once.

Noob Note: I had a difficult time finding the “Map To” so if you are too... This is one of the several small Windows that opens at the bottom of your screen after pressing **F5** and for me, it was on the far, far right, off my screen. I had to use my mouse wheel to scroll the Windows to the side to bring it into view and found it as a Tab on a Window with two other tabs. “Col” and “Nor” are buttons in the *Map To* Tab.

Another Noob note: In version 2.66, the “map to” equivalent is found in the texture window (under the tab “influence”) there you can adjust the color and normal settings. they are referred to by their full names.

- *What the buttons are for:* The “Map To” settings determine which various attributes of the material will be affected by our “cloudy texture.”
- “Col” refers to “color.” We turn it off here, because we don't want our surface to *look* “cloudy.”
- “Nor” refers to “surface normals.” This is the angle at which light seems to reflect from the surface. (The so-called “bump map.”)
- Experiment with the other “Map To” buttons, as well! For instance, “Disp” (for “displacement) actually causes a map to deform the geometry of a surface... but doing so only during rendering.

In this tutorial, as you will see, we are going to *combine* several techniques to produce a very rocky surface. Here, we are changing the way that the surface reflects light. Then, we'll actually deform the surface geometry.

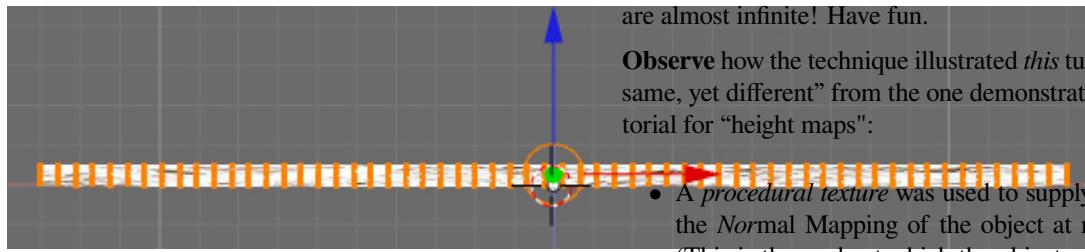
Turn the *nor* value all the way up, to 25 (that will give it a nice rocky look). Go into the *Shaders* tab and then change the “Diffuse Shader” (the top left selector, defaulting to “Lambert”) to “Oren-Nayer,” and the “Specular Shader” (the 2nd selector, defaulting to “CookTorr”) to “Blinn.” Adjust the *Rough* value in Oren-Nayer to 1.5 (rock is really rough). Also, set the *Spec* value to 0.01 and *Hard* to 25. Now we are ready to make some mountains!

- **Trivia note:** In computer graphics, a “shader” is a mathematical function, and they’re customarily named after the clever people who invented them. These functions determine exactly how the cloudy-texture will “map to” the attributes (*e.g.* “Nor...”) that we selected.



Making Mountains

Press **TAB** to go into edit mode and press the **AKEY** twice, in case you had any specific vertices selected. Go into the front view and press the “Noise” button (located under mesh tools in **F9**). You should see the vertices jump up a good bit.



If not, you may have scaled your grid up too much, so make it somewhat smaller. Depending on how high you want your mountains, press the *Noise* button to your required amount, though I pressed it 8 more times. The mesh doesn’t look very good right now, as it is not only blocky, but it is missing that last bit of random detail.

- **Important note!** The *mesh-editing button* named “Noise” causes a permanent change to the geometry of the object in the “Z”-axis, as provided by the texture. (The vertices actually *moved*, and their new position is permanent.)

- There is also a *procedural texture* named “Noise,” but that is just pure coincidence. The two are unrelated, both in what they do and in how they work.

- **Important note!** Those using Blender 2.5 Beta in order to get the height map to affect the geometry you will need to use the **displace modifier** instead.

If you are not in edit mode, go into it and select all the vertices. Press *Fractal* (in the same row as *Noise* under mesh tools) and enter a value of 15 in the random factor. Hopefully, your system can manage all these vertices: if not, then don’t do any fractal subdivide. If you want to, you can use a smaller value in the fractal box and do it several more times for more displacement. Be careful, though: too much randomness will cause vertices to separate, causing tears in the mesh that you will have to fix by hand.

Finally, tab out of edit mode and press *Set Smooth*. Then, if you want to make the mesh even more smoother, add a *Subsurf* modifier, with whatever level you want. (I’m going with 1 for faster render times.)

If you desire, you can scale up the mesh however big you want though for close-ups, you may want another level of subsurf or another subdivide. Then, apply colors, set up the lights, and render! A word to the wise, however: this process creates a huge file (about 986kb).

Discussion

You have just made mountains all in Blender, without having to generate height maps in The Gimp, or any other program. The next step is to go beyond the tutorial. Try using different noise sizes and noise basis, and even try using other textures like musgrave and marble. Or try using two textures and see the conclusion. The possibilities are almost infinite! Have fun.

Observe how the technique illustrated *this* tutorial is “the same, yet different” from the one demonstrated in the tutorial for “height maps”:

- A *procedural texture* was used to supply changes to the *Normal Mapping* of the object at render time. (This is the angle at which the object reflects light.) (The “height mapping” tutorial did not do this.)
- Then, the *mesh-editing button* (also...) named “Noise” was used to deform the actual geometry of the mesh. (This is the same technique used in the “Height Map” tutorial, although a different kind of texture .. an image .. was used.)
- The texture, once used for the height-mapping (deformations) was left in place, still mapped to the surface “Nor”mals. The result will be a *very* mountainous landscape.

Yep! The techniques shown in all of these terrain-modeling tutorials *can* be combined! And, they frequently are. Very often the most satisfactory results are obtained by combining several different techniques. You

could have used an image-map (from the “Height Mapping” tutorial) to deform the geometry, then used the “noisy Normal-mapping” idea from the first part of this tutorial, and even used smoothing. All at the same time. The choice is yours.

When using “Map To,” remember that a single material can have several different textures applied to it, each one mapped to the same or to different attributes. This mapping can even be *animated*.

Experiment! The sky’s the limit!

4.12 Landscape Modeling I: Basic Terrain

4.12.1 Introduction

I worked with a group to make a small animation for class and I was responsible for the environment modeling. I really liked the results of the process I used, so I decided to share it with others. By the end of this tutorial you’ll have the know-how to create your own flexible, realistic terrain utilizing multiple textures for different ground types. This tutorial assumes that you have the very basic understanding of using Blender (how to add/remove a mesh, how to change views, etc...) I use the following textures in this demo. Feel free to use them if you don’t have something on hand to use.

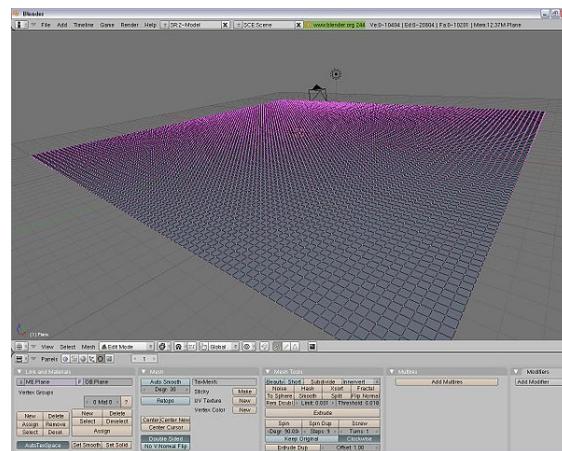


NOTE: This page uses a different grass texture for licensing reasons (sorry).

4.12.2 Creating the Canvas

The “canvas” that we’ll use for our terrain is an evenly spaced grid of vertices. Open a new project, delete the default cube, and **add a grid mesh**. Use whatever size you want. The more vertices you have, the more detailed and realistic your landscape will appear, but don’t get crazy with it since we’ll apply subsurfacing at the end to smooth it out. At the same time, you do want enough vertices to prevent sharp edges, and we’ll need them in the second tutorial when we cover texture stenciling. So consider how big you want your landscape and try to find a reasonable balance. I like to start with 100x100. The grid will be pretty small, so scale the entire thing up, let’s say by a factor of 20. This grid will be used to build our landscape by pulling hills and mountains out of it.

Noob Note: If you don’t want to have to delete the de-



The grid that will be used to build our landscape.

fault cube every time you open a new file, just delete the cube once so you have the blank window (only lamp and camera left). Then select *File - Save default settings* or hit *Ctrl+U*. From now on, each new project you create will start off without the cube.

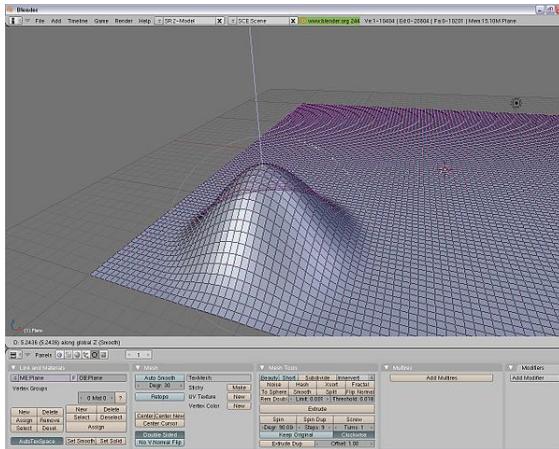
4.12.3 Molding the Mountains

The key to making good mountains is using the proportional edit mode (**OKEY**) and constantly adjusting the radius of influence. If you’ve already gone through the *Mountains Out Of Molehills* tutorial then this section will be familiar. One major difference is that in this tutorial I recommend rotating the 3D view around so you have a good view of all three axes instead of working in the front or sides view. Using the proportional editing tool affects multiple vertices, and it helps to see what effect your changes are having as you make them.

- Go ahead and turn on proportional editing, either by pressing **OKEY** or clicking on the grey ring on the 3D View header. You have to be in Edit Mode to select this option. Once proportional editing is enabled, the ring will appear orange and a new dropdown menu will appear next to it with different falloff styles. Select Smooth if it is not already selected.

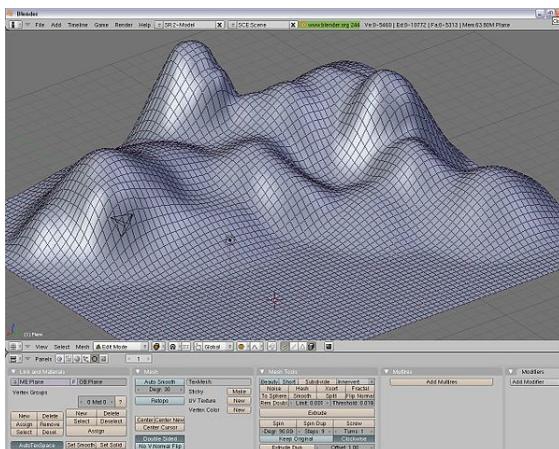


- Select any random vertex and grab it (**GKEY**). You will see a ring around the vertex you are grabbing. This is the radius of influence, and only vertices inside this ring are affected by the transformation. If you are doing this in the orthographic view from the front, side, or top, then this will be obvious. But if you’re at a view where you can see all three axes, then it may be less obvious.



The first mountain created using proportional editing.

- Restrict movement to the z-axis (**ZKEY**) and translate the vertex upward. Throughout this entire process you ONLY want to translate along the z-axis. If you start moving vertices in the x or y directions, things become distorted and you get some nasty creases. Play around with adjusting the size of the radius of influence (**Mouse Wheel**) to get steeper or flatter hills.



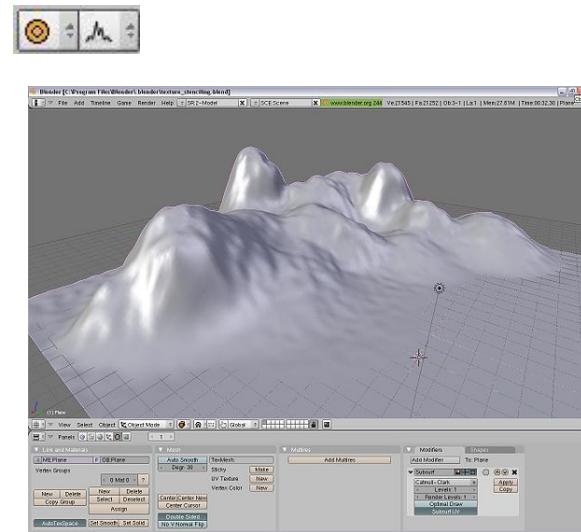
A nice group of hills

- Keep repeating this process with different size radii and different heights until you have a decent mountain range, but leave an area flat. We'll be using that spot later in the second tutorial. Don't be afraid to occasionally translate some vertices down instead of up to create depressions in the hills. Remember, variety is the spice of life. Very few things in nature are naturally geometric, so mix up your hills and especially make sure they overlap. How often do you see a nice, smooth hill all by itself in nature?

Note: You'll notice in my screenshot that I have reduced the size of my grid. For simplicity's sake, I didn't feel

like filling an entire 100x100 grid with mountains since this can take some time.

- Well now, that's looking pretty good! Now, there's one problem with our hills so far. They're too smooth! Let's bumpify them a little. Change the falloff type from Smooth to Random.



Rougher hills look more realistic

- Select a single vertex and grab it (**GKEY**). We're still working on the z-axis only, to restrict your movement with the **ZKEY**. Now when you move the vertex up and down, all vertices in the radius of influence will also move but with a random falloff instead of smoothly. It only takes a little movement to get the effect we want, so something around 0.5-1.0 is enough. Mix up moving up and down with different vertices, again to add variety to the scene.
- Once you have your landscape the way you like it, add a subsurf modifier under the Editing tab (**F9**) and select Catmull-Clark. This will smooth out your terrain a little so that it's not too rough. Given the number of vertices you already have, it's not necessary to have a higher render value than 1 unless you just REALLY want it to be smooth, but I don't recommend it. Land is supposed to be bumpy and rocky, we just don't want sharp edges.

Noob note: If you're making mountains using Random Falloff and the peaks stick up too much: in Edit mode, select the points in the area around the peak using circle select, then press **WKEY** and click 'Smooth' until you're satisfied (or, in the Editing tab (**F9**), click the 'Smooth' button).

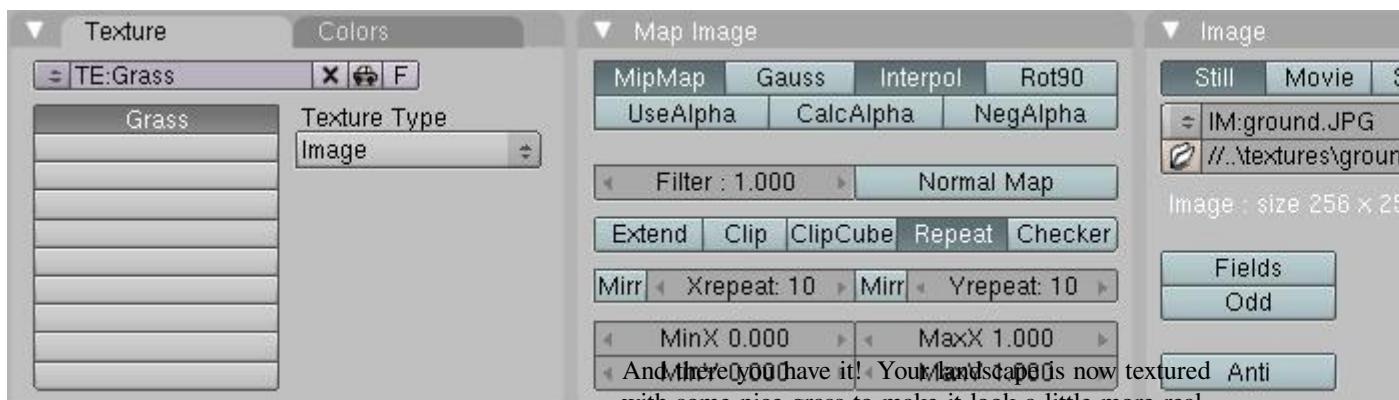
Useful Tip Proportional editing can be used on multiple vertices simultaneously. This is especially useful if you're trying to create a cliff face or a river bed. Use the box tool

(BKEY) to select a group of vertices and then translate them. Keep in mind that the size of the radius of influence determines how many vertices around **EACH VERTEX** will be influenced. So suppose you have a 5 vertex radius, that means that 5 vertices all the way around your selected region will be influenced.

4.12.4 Texturing the Terrain

Alright! We've got some pretty nice hills now! But there's still a few problems. Hills shouldn't be white, and hills shouldn't be SHINY! Let's dress them up a little, shall we?

- With your landscape selected, go to the Shading Panel (**F5**) and add a new material.
- Under the Shaders tab, drop the Specular value to 0.
- Go to the Texture tab (**F6**) and add a new texture.
- In the Texture Type drop down menu, select Image.
- Two new tabs will appear. In the Image tab, click Load and load a texture from file.
- In the Map Image tab, increase the Xrepeat and Yrepeat. Depending on the size of your terrain and the image that you use (please use something that tiles!), these values will vary. I've used 10 for each in this tutorial.

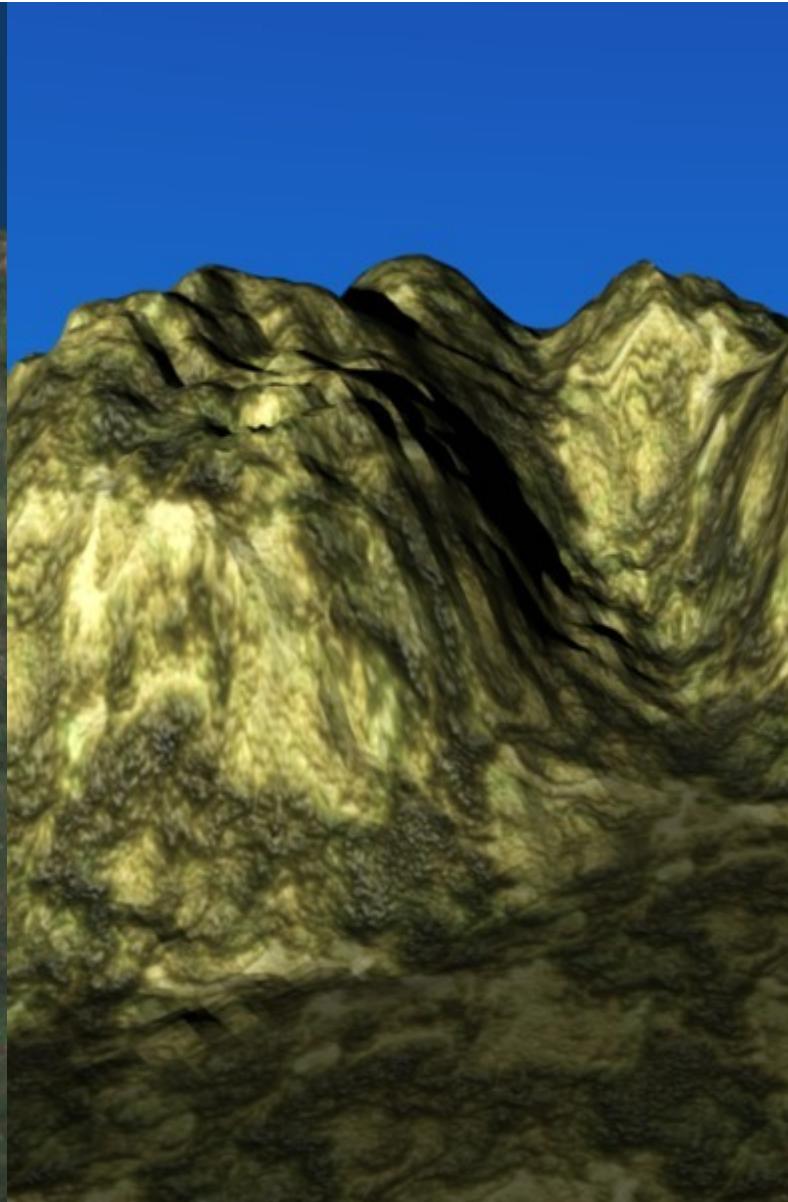
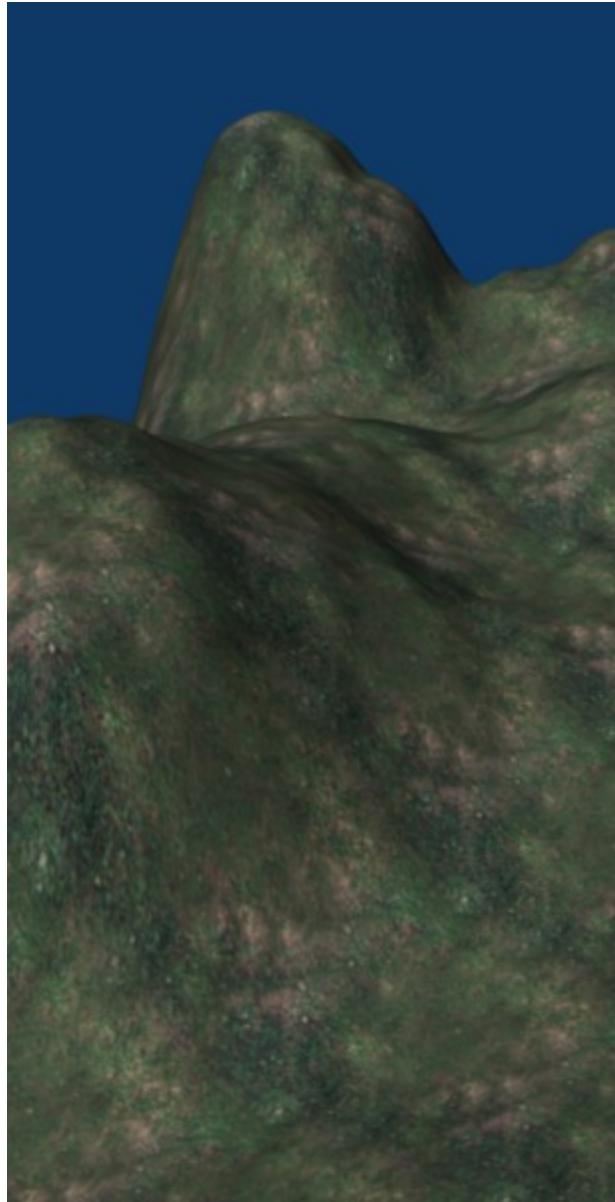


(Noob note: Render the scene (**F12**) to see the applied texture) **(Noob note:** A quicker way to do this is by using the render preview tool, 3d view window - render preview, **View--> Render Preview** or press **SHIFT--> PKEY**) **(Noob note:** you may also select shaded in the Viewport Shading menu next to where you select object mode or edit mode to see your texture on your mountains without rendering, however it does slow your computer down some which could make editing frustrating. So only do this if you want to see what your textures look like, then switch back to solid for more editing.)

- Finally, let's do something about the lighting. Go to Object Mode (**TAB**) if you're not already there

and select the lamp. Choose the Shading Panel (**F5**) and then click the icon that looks like a light bulb to display the Lamp buttons. Change the lamp to a sun and up the energy to 1.5. You may also need to increase the distance if your terrain is large, or rotate it around if you don't like where it's pointing. The dashed line coming out of the light is the direction. Play around with different angles and energy values for the sun to get different times of day in your scene.

NOTE If you went with the suggested 100x100 grid, the rendering process could definitely take some time depending on your system, especially if you have ray tracing and shadows enabled. To boost the rendering speed, go to the Scene panel (**F10**) and under Rendering, disable the buttons that say **Shadow** and **Ray**. Also note that because we've created some hills, your camera may now be under the terrain. Switch to the camera view (**NUM0**) to see what your camera sees and move it if you need to.



Join us next time as we explore how to make the landscape look even better using texture stenciling!

4.12.5 Reader Contributions

I'm mainly doing this so the hills look better and more realistic. Reading this will consume more time than doing it. I'm making everything clear for beginners. It will only take about 8 minutes more for your hills to end up like this:

1. First step to achieve it is to switch from the default lamp to Sun.
 - (a) You do this by clicking on the default and go to the shading tab (F5)
 - (b) Click on *Sun*.
 - (c) Go to *rotate manipulator mode* with Ctrl Alt R.
 - (d) Rotate the sun until the dotted line is in your desired position. That is where the main energy will go. It will differ if you want to achieve the different time of day. You can make the distance greater or less with *dist..*. I kept mine at default 30.
2. Now let's make more realistic, paler sunlight.
 - (a) In the RGB slider, set R for 1, G for 1, and B for .848.

- (b) Set energy for 1.63. This will all differ for different times of day, so set your east and west in your head, and the later into the day, the further the sun to the west and the more orange. For midday, keep the energy on 1.63 and put the sun right above your hills. I set mine for earlier in the morning.
3. Now click on the picture of a globe to change the background.
 4. Click on the *blend* button for a more realistic looking sky. On the left of the World toolbar, that will be the color of lower down in the sky and the right sliders will be the color of the top of the sky. Naturally, the top should be darker blue than the bottom.
 - (a) It will differ for different times of the day. So you might want something different from this. But I set up mine for the morning using the settings below:
 - The left HoRGB sliders to 0.50, 0.68, and 1.
 - The right ZeRGB sliders to 0.11, 0.25, and 0.66.
 5. Now for the texture. *This* won't differ for the times of day, but it will differ for what type of landscape you want.
 6. Download a nice texture from google or an artists website.
 - (a) Go back to Blender, go to Texture buttons (F6)
 - (b) Click add new
 - (c) Select *image* for the texture type
 - (d) Go to the *image* toolbar, and upload the texture. Blender will only allow you to upload from the Blender documents, unless if you click on the up and down arrows on the top left of the screen and choose the place where you saved the texture (or picture).
 - (e) After uploading, the X and Y repeats should be smallish, like 6x6, so the changes aren't noticeable. It will look ugly in the preview, but it will look nice after its wrapped.
 7. Render, and 1 minute later, voila! Those are nice looking hills!
 8. Now boast in front of your friends!

Noob Note: You need to have your landscape selected if you are in object mode to change the texture, otherwise it will change the texture of the world.

4.12.6 Reader Contributions 2

You can have an even better result if you use the texture to “bumpmap” the mountains.

1. Press **F5** until you are into Material Buttons
2. Select the *Map to* tab
3. Click on *Nor* once (the *Col* option must stay selected as well)
4. Slide the *Nor* slider to 5 or more (might differ depending on texture size and repeat options)
5. Render with **F12**

4.12.7 Reader Contributions 3

Alternately from making the hill model manually, you can use a program like **L3DT**. It uses various algorithms to generate very detailed and realistic terrain heightmaps. It also lets you edit them in a more intuitive way than Blender does. After you have L3DT make your height map, you can export it as a .x file, which you can import into Blender. **Noob Note:** That can be done by going into File->Import->DirectX(.x). And there you have a very realistic landscape mesh with a lot less work. (Noob Question: I want to use Unity3d game engine to create a game, but i want to use blender to create landscapes. To create massive world landscapes, does anybody have any tips? as Unity limits any mesh to 65000 vertices, which i've already passed with my method without doing half the map.)

-If you are using unity 3d just use it for the terrain. Download the terrain toolbox and just use .raw images. Or simply generate from there and save. 65,000 verts is plenty btw. If you insist on using blender for the terrain then model a high poly terrain fist consisting of around 1 mil+ verts. bake the normal map and ambient occlusion decimate your terrain to around 10k or w/e looks good for the game. Then bam you got 10k terrain that looks like 1mil. But to be honest blender terrain texturing is a pain compared to unity. Anyhow hope that helps.

4.13 Landscape Modeling II: Texture Stenciling

4.13.1 UPDATED FOR 2.7 USERS

This is a continuation of the previous tutorial, **Landscape Modeling I: Basic Terrain**. In this tutorial, we will make our terrain look even better by using some texture stenciling to add multiple textures where we want them in the landscape. This tutorial assumes that you have a basic understanding of how to use Blender (how to add/remove a mesh, how to change views, etc...)

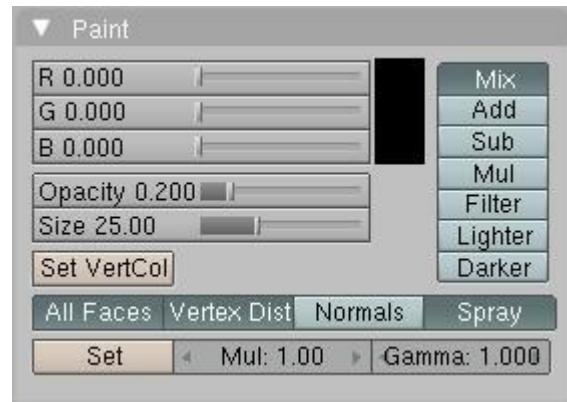
4.13.2 Creating the Stencil

The landscape from the previous example looks pretty good, but the entire thing has the same texture, so it doesn't look very natural. Let's add some rocks to those hills. If we just add a second texture to our material, it will completely cover the previous one. What we want is to have the first texture only show in certain places and the second texture cover the rest. In order to do this we'll have to create a stencil, which is like a mask that determines where textures appear on the material. The stencil is a black and white image, similar to a heightmap, except the intensity of each pixel determines how much of the next texture will appear (black = 0%, white = 100%). I highly recommend viewing [this tutorial](#) for a more in depth description of how stencils work. If you think you have the gist, then continue on here. [Note: Link is broken now] We're going to create our own stencil to determine where we want rock in the landscape by "painting" on the object. ***IMPORTANT*** The next few steps will be performing temporary modifications to your scene, so be sure to save your file before you continue so that the changes will not be saved!! Also, if you have a subsurf modifier on the landscape, remove that at this time as it drastically slows down the following process.

- Select the landscape object and switch from "edit mode" or "object mode" to "**Vertex** Paint mode". This is found in the Mode drop down menu, on the 3D View header. This mode lets you paint the object, and thereby change the color value of each vertex. When in Vertex Paint mode, you'll notice your object change color to a pixelated version of the texture that is applied to it. This is because with that texture applied, each vertex will be drawn with the colors you see.

Noob Note: Vertex Paint mode is in the drop down menu that contains Object mode and Edit mode. Just select the object while in Object mode, click the drop down menu, and select Vertex Paint mode from the list.

- Switch to the Editing menu (**F9**) and you'll see a new tab called Paint. Notice the sliders labeled Opacity and Size. Opacity determines how much to blend the selected color with the existing colors when you paint. So 0.2 means that when you click, you'll get 20% of the selected color mixed with whatever is currently there. Over on the right are different options to combine to colors. We want to mix, so as we paint the colors will mix together and give us a nice smooth blend. Size is the size of your cursor while painting, and thus how many vertices are affected. Keep in mind that does not change as you zoom in and out, so a size of 10 can actually paint more pixels if you're zoomed way out than a size of 30 if you're zoomed way in.



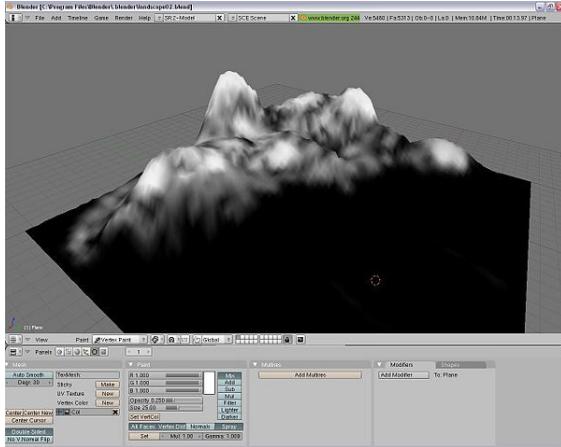
- Change the color to black and click the button that says Set VertCol (in 2.48 version, that button is below the color square and named 'SetVCol'). This will change the color of your entire object to black. You'll notice that the object is not lit in Vertex Paint mode, so when the entire object is black it can be difficult to see where your hills are. You'll have to move the camera around to see the shapes.

[Another newbie recommends: if you switch to the "object" buttons (f7) and turn on "Wire" under "Draw Extra" in the "Draw" panel/tab you should see a wireframe on top of the vertex colors you are painting, which makes it much easier to see where the hills are. Remember to switch this back off when you have finished painting.]

- Now we're going to paint the places where we want rock to show through. Change the paint color to white. It's less likely that grass will grow on steep slopes, so start painting the tops and edges of the steeper hills white by holding down **LMB** and dragging your mouse (think MSPaint). If you didn't heed my earlier advice and still have a subsurf modifier in effect, the painting will be very choppy, so remove that now.
- As you're painting, try to make some spots pure white by going over them again and again, but don't make any one area of white too large as this will make a huge area be all rock, and we're trying to blend two textures.
- This process can just be trial and error getting the landscape painted the way you want. Once we apply our stencil later you may decide there's too much rock in one area, or not enough in another and go back and change it. A few things to keep in mind:
 - 20% opacity means that most of the original texture (grass, in our case) will show through, so the rock won't be very noticeable if at all. You'll have to go over the same spots a few times to increase the intensity.
 - You don't want a dramatic change from grass to rock, so be sure to blend white areas with

black. That's why we're using the 20% opacity instead of just bumping it up to 100%.

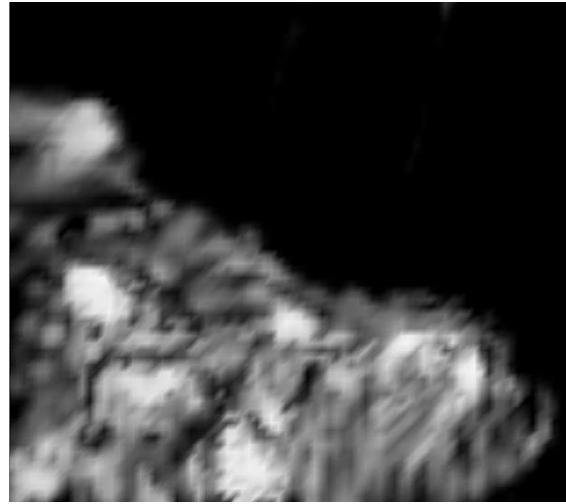
- Make your patches random and spotty. This will end up creating a more realistic effect once we combine the textures.



Painting the stencil onto the landscape

- Once you're done you should have something that looks like this. Notice the mixture of white and black in some places, how it doesn't just go from pure white to pure black as you move down the hills. This will make a more natural blend and it will cause random rocky areas mixed in when the grass. Now we need to turn this into our stencil.
- Go to the overhead view (**NUM7**) and make the projection orthographic if it's not already (**NUM5**).
- Zoom in/out (**Mouse Wheel** or **NUM+/-NUM-**) until the plane almost occupies the entire window.
- Move the 3D cursor and any lamps or cameras away so that they're not over the landscape and take a screenshot.
- Open your favorite image editor (I'm a traditionalist, so I still like MSPaint), and cut out the image of landscape. This is now your stencil. Save it under your favorite format.
- Re-open the saved version of your landscape to undo the changes we made to create the stencil.

So why did we make our stencil this way? This allowed us to actually paint on our terrain so that we ensure we get the rock exactly where we want it. Using this method, you can customize your stencil to any object, as long as it has enough vertices. For example, if we just had a flat plane made of 4 vertices, this technique would not have worked because we could only paint the corners. That's why in the previous tutorial I recommended using a high (but not too high) number of vertices in your grid.

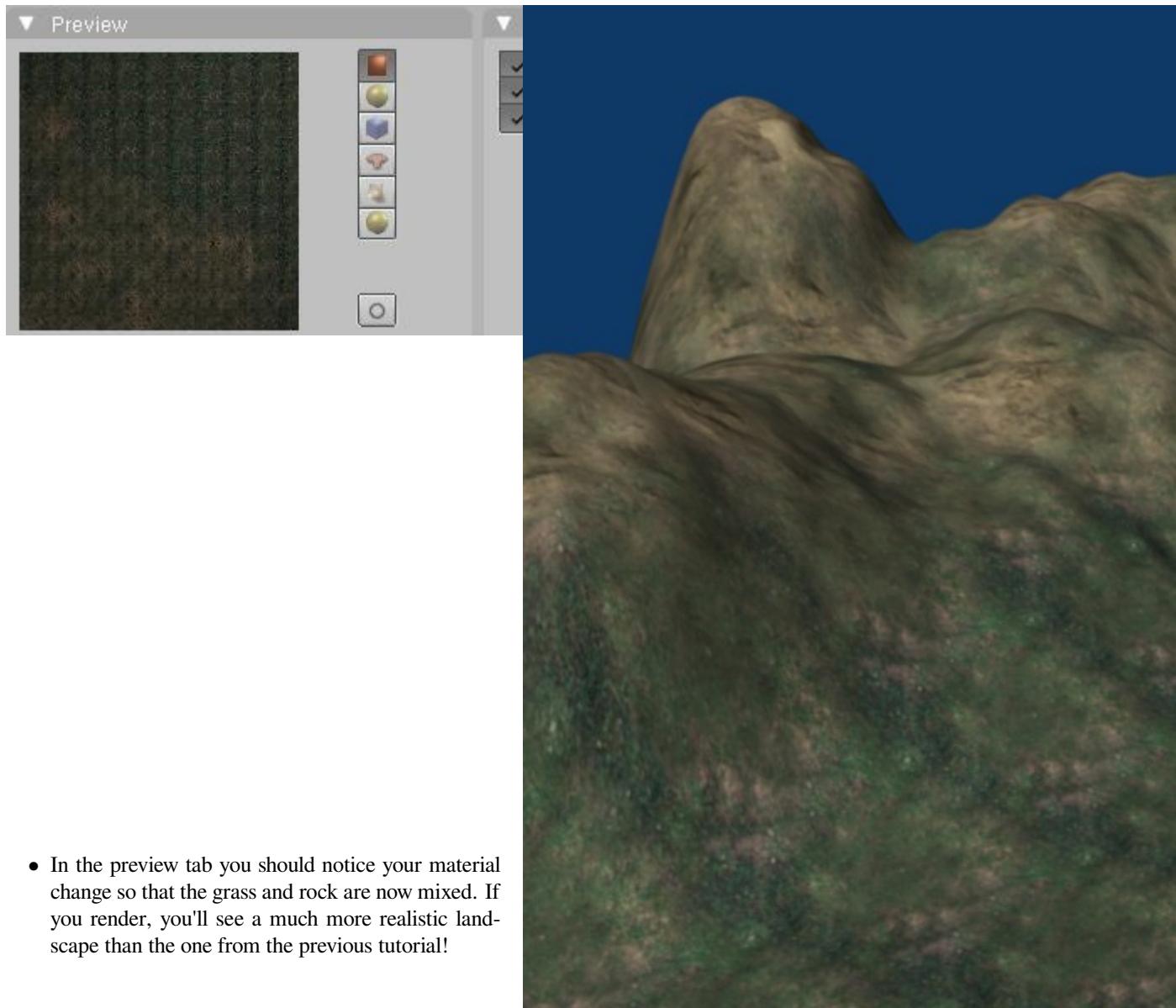


The stencil after its been cut from your screenshot

4.13.3 Applying the Stencil

So now we need to apply the stencil to the landscape to mix our grass and rock textures together.

- Return to either Object Mode or Edit Mode (**TAB**), it doesn't matter which.
- Go to the Texture menu (**F6**) and add two new textures in the material (you should already have the grass texture).
- Make texture 2 (the first new one you added) an image and load your stencil from file. Do not repeat this in the x or y directions. We want it to map to the entire object.
- Make texture 3 an image and load your rock texture from file. This one you do want to repeat, just as you did the grass.
- Now return to the Materials menu and select your stencil texture. It's a good idea to name your textures, materials, objectives, etc... so that they're easily identifiable.
- With your stencil texture selected, switch to the tab labeled MapTo. Deselect Col (which maps the texture to the color), and select Stencil and No RGB. Stencil will treat this image as a stencil, and No RGB treats it as a black and white image. If you didn't select this second option your stencil wouldn't work.



- In the preview tab you should notice your material change so that the grass and rock are now mixed. If you render, you'll see a much more realistic landscape than the one from the previous tutorial!

4.13.4 Adding Snow

Since you already have some nice mountains, why not add some snow to them? Adding snow is easy. All you have to do is add a fourth texture AFTER all the others. Make this texture the same as your stencil.

- Make sure you're in Shading (F5) > Texture Buttons (F6) > Textures.
- Click on an empty panel beside Texture Type
- Click on the little box to the left of Add New
- Choose your stencil from the menu (hint: it helps if you name your textures)
- Render and see the results.

Noob Info: Be sure that for the rock - texture *col* is selected because we want to color the landscape with the rock-texture! Otherwise it doesn't work. *Select Rock-Texture -> Map To -> col*

If you made your stencil properly, you should have some nice, snowcapped peaks. It's not much, but it's enough

for a quick fix. If it isn't even, you probably didn't make your stencil properly (meaning pretty much all white on the peaks, and not much anywhere else). Of course, there are better ways to do this, ways I wouldn't know about; I discovered Blender merely three days before the time of writing. I leave it up to you to see the effect: I only have so much room and bandwidth for pictures!

This effect works by saturating the underlying color with white, depending on the amount of white in your stencil. A light gray will saturate it only a bit, whereas a bright, 100% white will cover it very clearly and efficiently. If you look closely, you'll see that the light gray on the stencil DOES make a difference, but it's not immediately visible and requires lots of fidgeting with the Render Preview and texture options. You might also try fiddling with the Stencil and NoRGB option, although it worked fine for me with both turned off (I use Blender 2.48a)

Some challenges:

- Try to change the color of the snow. Hint: RGB
- I've yet to do this, but try making snow out of a heightmap.

4.13.5 Multiple Stencils

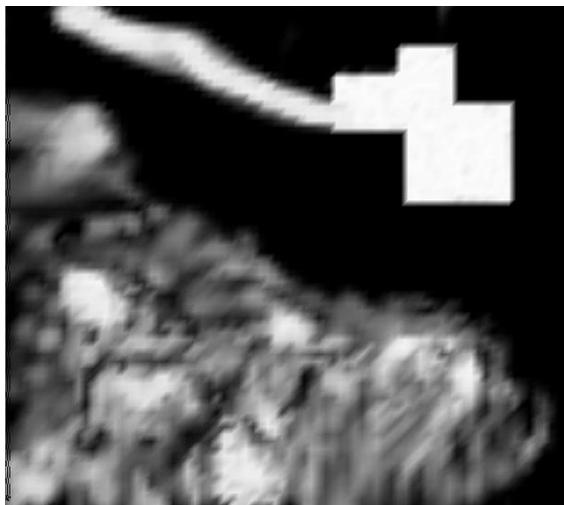
Remember in the first tutorial when I said to leave part of the landscape flat, because I'd be using it later? Well it's time to use it. I want to eventually use the landscape as the backdrop for a military base, so let's texture the ground around where the base will go to give it a dirt ground and road leading to it. ***IMPORTANT*** Just as before, the next few steps will be performing temporary modifications to your scene, so be sure to save your file before you continue so that the changes will not be saved!!

- Switch to the Vertex Paint mode in the mode menu on the 3D View header.
- Go to an overhead view (**NUM7**) and make the projection orthographic (**NUM5**)
- Once again, paint the entire object black by selecting the color black in the Paint tab and clicking on Set VertCol.
- For the next few steps, you may need to **TAB** back and forth from Vertex Paint mode to Object mode to make sure you're painting in the correct area. Make the opacity of the painter 1.0 and the size 10 so that we can work very precisely.
- Paint an area over the flat terrain in the shape of a generic military base, like you see here.
- Reduce the opacity back down to 0.2 and increase the size to 20.



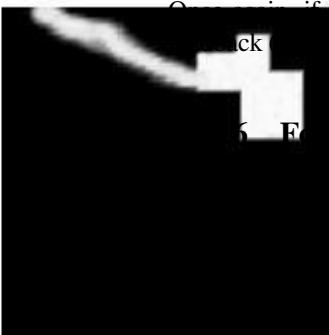
The stencil for our future base

- Paint the road leading away from the base. We've reduced the opacity and increased the size so that the road will blend more with the landscape around it instead of being more defined like our base.
- Once you have your stencil looking the way you want, take a screen shot and save it. Then revert back to your previous environment (before you started painting on it).
- Go to the Texture menu (**F6**) and add two new textures.
- Load your stencil into the forth texture (the first new one), and the dirt image into the fifth texture. Once again, remember to set the x and y repeat for the dirt texture.
- In the Materials menu (**F5**) select your new stencil and go to the MapTo tab. Deselect Col again, and select Stencil and No RGB. Notice the new preview.
- Wait a second, where's the dirt path?!? Here's the problem and the reason I've included this section: stencils are cumulative. That is, the first stencil defines which portions of ALL successive textures will be drawn, including other stencils! In order for our new dirt path to show up, we need to make sure it's part of the first stencil.
- Open your first stencil (the one you used to add rocks to the hills) in your image editor and combine the new stencil with it. The result should appear somewhat like this. Be sure that when you combine your dirt stencil with your rock stencil that the dirt stencil remains the same size and in the same place. That is, you want the two to be perfectly aligned.
- Reload your first stencil on your material, the one we used for the rocks. Now the preview should look



Both stencils combined into one

the way we want it to, and if you render you will see your new dirt area on the flat part of the landscape. You can add even more stencils the same way. Here is how all of the textures blend together:



Open up a discussion if you have any issues with the tutorial, or click the like or negative, drop it in the discussions.

6. For 2.7 blender users:

+ and the material

grass texture at the default

next two textures , the stencil and the rock, soil etc texture as stated in the above tutorial.

NOTE I had to replace the grass texture in the above image since I could not find the licensing information for it. I apologize for the inconsistencies in the diagram that result from this.

Well there you have it! This is a pretty useful method of texturing complex objects. Check out our new landscape and how it compares to the previous, boring, single-textured landscape!

- go to the stencil texture
- in the influence panel, uncheck col
- check 'RGB to intensity' button
- check stencil
- others everything as their default values are good

-For me, with settings above, the rock texture was not very visible. It may help to play around with the color intensity settings in the influence panel (increase in the rock texture, and decrease in the grass texture) -

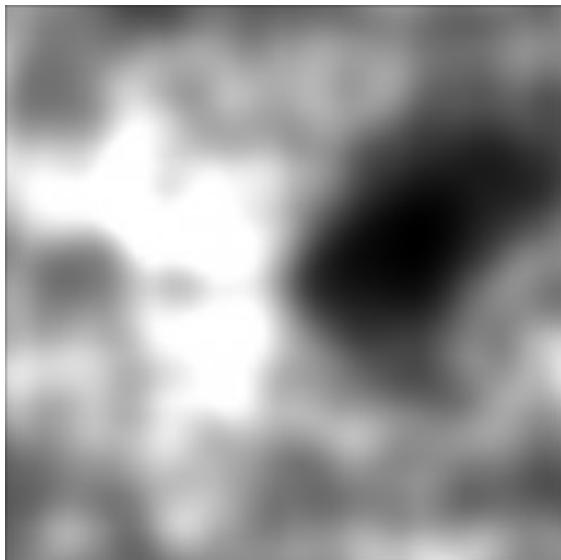
4.14 Landscape Modeling III: Exporting as a Heightmap

In this tutorial, I will show you how to export your beautiful terrain to a heightmap that can be used in most 3D graphics engines. The benefit to doing this as opposed to just exporting your mesh is that many engines have a special process for dealing with terrains as opposed to regular meshes where it divides the terrain up into different sections so that it can subdivide regions closer to the camera more for greater detail, and also cull (not render) regions outside the camera's field of view, saving precious processing time. This structure is called an **oct tree** and is a highly optimized way to render large meshes such as terrains.

Before I begin, I must give credit to the [Creating a Heightmap from a Plane](#) tutorial on the [Blender wiki](#). This is how I learned to do this trick, and most of what I will cover came from this tutorial. I'm simply including it here with the other landscaping tutorials I've written for convenience.

4.14.1 A Word About Heightmaps

I will briefly cover what a heightmap is and how it's useful for those who may not know. If you are familiar with heightmaps, you can skip ahead to the next section.



An example of a heightmap

A heightmap is a grayscale image that uses various shades of gray to represent different elevations across a map.

Since the images are 8-bit (with the exception of some RAW formats that are 16-bit), you have 256 different shades of gray, ranging from pure black (0) to pure white (255). Black represents the lowest elevation on the map, while white represents the highest. Many 3D graphics engines already have functions to read in a heightmap and generate terrain from it. The way it does this is it creates a grid of vertices (like we are going to do in Blender in just a moment), and uses the heightmap to determine the elevation of each point on the grid. The more intense the pixel color (the closer it is to white), the higher the elevation of that vertex. In most cases, if the resolution of the heightmap is smaller than the resolution of the terrain, the engine will interpolate the vertices in between those set by the heightmap. For example, if your heightmap is 256x256 and your terrain is 1024x1024, your heightmap will determine the elevation of every fourth vertex, and the three in between will be interpolated.

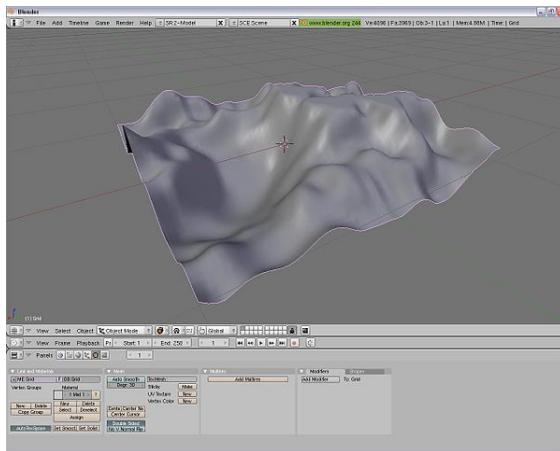
Note that these values do not represent the absolute height of any given pixel, but rather the height relative to the rest of the map. That is, your heightmap does not represent a landscape with altitudes ranging from 0 to 256. When importing a heightmap, you can specify the minimum and maximum altitudes for the map, and the whole thing gets scaled. For example, suppose you decide the minimum is 1000 feet, and the maximum is 5000 feet. When the terrain is rendered, black pixels will represent 1000, white pixels will represent 5000, and a pixel that is exactly in between black and white (128) will represent 3000 feet (half of the sum of min and max). This makes heightmaps relatively flexible.

Unfortunately, as you may have already noticed, there is a limitation of using heightmaps. Since you only have 256 shades of gray, you only have 256 possible elevations. This causes a problem. No matter how precise your vertices may be in the mesh that you model, no matter how smooth you may have it looking, each point will get rounded off to an integer value between 0 and 255. When used to render a terrain, this can cause a "stair step" effect as the terrain goes from one discrete elevation to the next instead of making a smooth transition. But there are ways to avoid this, which I will cover later.

I apologize if you're scratching your head right now saying "Huh?" For a further explanation and example, check the Wikipedia page on [heightmaps](#).

4.14.2 Creating the Material

To begin this tutorial, I will assume that you already have a landscape that you want to create a heightmap for. If not, read my first tutorial, [Landscape Modeling I: Basic Terrain](#), to quickly create something. **WORD OF CAUTION!!!** If you've already created a landscape, try to remember the exact size of the plane that you used. If you're about to make one, it's okay to scale the plane to have a larger surface to work with, but REMEMBER how



The terrain that will generate the heightmap

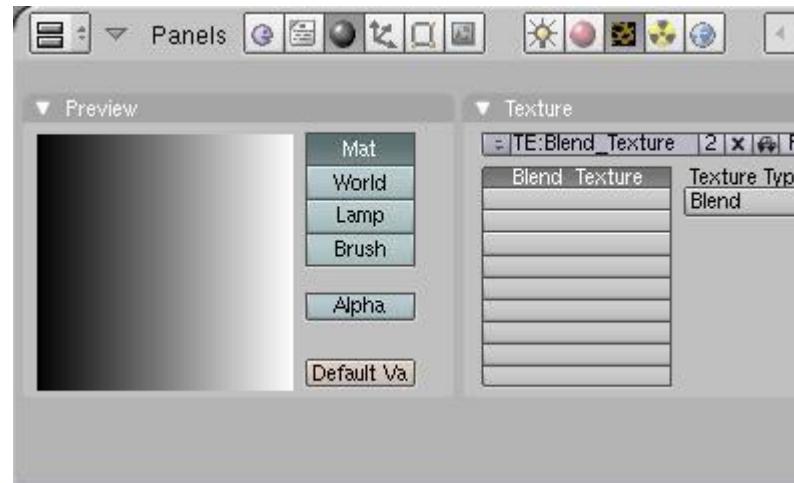
much you scale it by. This will make things easier later. I recommend just working with the default 2x2 plane or grid and zooming in on it. You may also want to backup your scene since we'll be changing a few things

To create the material we need, do the following steps:

- Remove any existing materials that may be applied to the terrain.

Note: to remove a material, I think you need to go to the links and pipeline window in the shader section and click the little “X” underneath the words “link to Object”

- Add a new material in the Shading tab (**F5**).
- Go to the Textures tab (**F6**) and add a new texture. This should be the only texture on your material.
- From the Texture Type drop down menu, select Blend.
- In the same window, find the Colors tab and select Colorband. This will let us define the blend pattern, which by default fades from black with full transparency to a solid cyan. Why cyan? I have no idea. It's an odd default.
- Select the black color by clicking on the left side of the colorband. You should see a little black and white bar selected.
- Increase the Alpha all the way up to 1 by adjusting the slider labeled “A”.
- Select the cyan color and change this to a solid white.
- What you have now should look like my colorband below.



- Return to the Materials tab (the red ball icon next to the cheese looking icon).
- In the Material pane, select the Shadeless option to disable lighting on the material.
- To the right, find the pane labeled Map Input. In the bottom of this pane is a grid that by default reads:

XYZ
XYZ
XYZ

- Set all three rows to Z so that the texture will map entirely to the Z coordinate.
- In the preview window to the left, if you select the sphere or cube you should see that it goes from black on the bottom to white on the top. That's the effect we want for the terrain. Remember, in a heightmap, black represents the lowest point and white the highest.

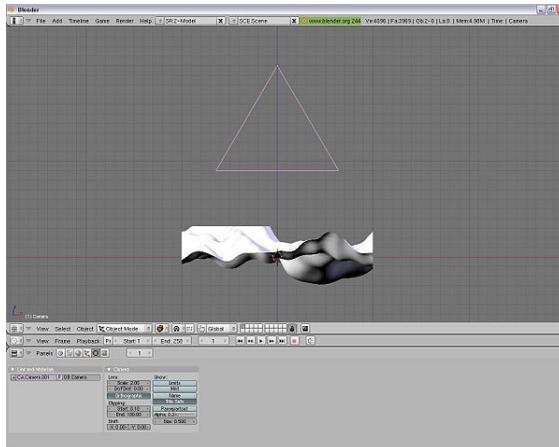


- Finally, switch back to the Editing tab (**F9**) and click the Set Smooth button to enable Gouraud shading so that everything appears smooth.

4.14.3 Setting up the Camera

That's all we need to do for the terrain itself. Now what we're going to do is setup the camera in such a way that when we render the scene we'll get an orthographic projection straight down on the terrain so that we can save it as our heightmap.

- If you still have any lights or cameras in the scene, delete them.
- Switch to the top view (**NUM7**) and place the 3D cursor at the origin (0, 0, 0) by left clicking near the origin, hitting **SHIFT+SKEY** and selecting *Cursor to Grid*. (An easier way is: hitting '**SHIFT+SKEY**' and selecting '*Cursor to Center*')
- Add a new camera, *Space->Add->Camera*. This will add a new camera to the scene that is looking straight down.

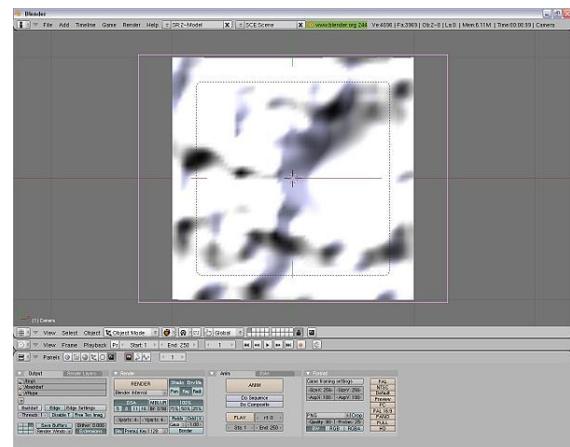


The camera needs to be above the terrain

- Switch to a side view (**NUM1** or **NUM3**) and move the camera up the Z axis so that it is above the highest point in your terrain. Even though the view is going to be orthographic, and therefore distance doesn't matter, the entire terrain still needs to be in front of the camera.
- Now, with the camera selected, go to the Editing tab (**F9**) and select the "Orthographic" button. This will change the camera to an orthographic projection which basically ignores the z-coordinate for all vertices (except for determining which pixels should be in front of others).
- Just above the orthographic button is a value labeled "Scale". Remember earlier when I told you to remember how big your terrain is? This is why.
- Change the Scale of the lens (default: 6) to match the size of your terrain. If you left the plane the default size, this should be 2. If you scaled the plane

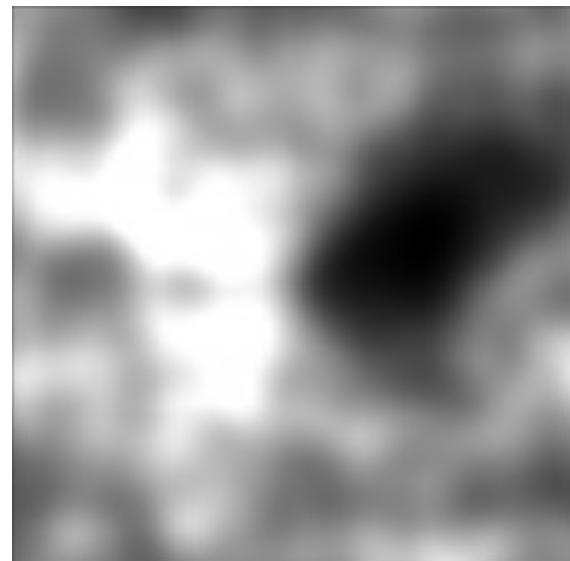
up by a factor of 20, as suggested in the "Landscape Modeling I: Basic Terrain" tutorial, this should be 40 (the plane is initially 2 units square).

- Now go to the scene tab (**F10**) and under Format, change the SizeX and SizeY to be the size you want your heightmap to be. For most graphics engines, this is required to be either a power of 2 (2^n) or one more than a power of 2 (2^{n+1}). So try something like 256 (or 257 if you need it to be 2^{n+1}).



What the terrain will look like from the camera view

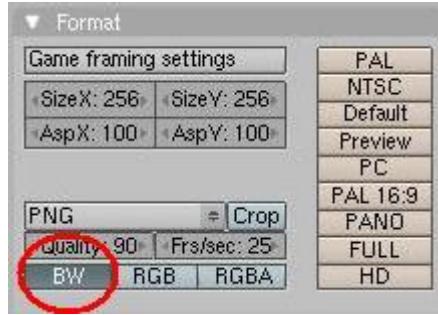
- Now, if you switch to the camera view (**NUM0**) you should see a "flat" square that completely fills the dotted box which represents the camera's field of view.



The rendered image will be your heightmap.

- Render the scene (**F12**). You should be looking at your heightmap. Hit **F3** to save the rendered image and you're good to go!

Since this image has no color, and some engines require that your heightmap be a grayscale image (only one color channel instead of three), select the BW button in the Format pane before rendering. This will render it as a grayscale image instead of a color image.



4.14.4 Avoid Stair-Stepping

As mentioned earlier, exporting the heightmap as an 8-bit image (Blender doesn't support any 16-bit formats that I'm aware of) has the drawback that you only have 256 discrete height values which can cause a stair-step effect when the terrain is subdivided. Ideally, you could use an application that can export the image in a 16-bit raw format, such as Terragen. But this is a Blender tutorial, so we'll not go into that. =P

Reducing the resolution

One way you can "smooth" out the rendered terrain is to actually reduce the resolution of the heightmap. This may sound counter intuitive, but bear with me. Suppose you have a heightmap that is a simple blend from black to white going from left to right. If your heightmap is 1024x1024, you will get 4 columns of each height since you only have 256 different values. In other words, your grayscale will look something like this:

00001111222233334444...

00001111222233334444...

00001111222233334444...

00001111222233334444...

00001111222233334444...

If you try to render terrain from this, you'll get 4 points with 0 altitude, 4 with 1, 4 with 2, and so on, creating the stair-step effect.

Now suppose you make the same heightmap, but reduce its size to 256x256. Now you'll have 1 column for each height, and your grayscale will look something like this:

012345678...

012345678...

012345678...

012345678...

You may be thinking, "But then my terrain will be smaller, or less detailed." But it turns out the opposite is actually true. Consider applying the above heightmap to a terrain that is 1024x1024. The values from your heightmap will be applied to every 4th vertex, and the ones in between will be interpolated. So instead of having something like:

0.0 0.0 0.0 0.0 1.0 1.0 1.0 1.0 2.0

which is what you would get with the 1024x1024 heightmap, you'll actually get something like this:

0.0 0.25 0.5 0.75 1.0 1.25 1.5 1.75 2.0

Where the integers are the values from your heightmap and the fractions are interpolated values. This is assuming that whatever engine you are using will interpolate the values between heights read in from the heightmap.

At the present time, this is the only method I have verified that helps reduce the effects of stair-stepping. As I try different techniques I will update this page with them. For example, I know that Adobe Photoshop can export images in 16-bit raw formats, but I haven't tried it yet so I don't want to suggest it as a solution.

4.15 Bump Mapping

Bump Maps are textures that store the relative height of pixels from the viewpoint of the camera. The pixels seem to be moved in the direction of the facenormals, either in direction to or away from the camera. You may either use greyscale pictures or the intensity values of an RGB-Texture (including images).

Bump Maps are easy to apply. They work well on flat surfaces, only to some extent on curved surfaces. On curved surfaces it is more easily noticeable that you don't create real 3D structures. The visible effect depends on factors like lighting, specularity of your material, camera angle, distance and so forth.

- Add a new texture to your object.
- Activate the *Nor* button in the *Map To* panel.
- Set the depth of the bumping with the *Nor* slider.
- Use the texture type *Image* and load your bump map.

Bump maps should contain hard transitions between black and white. A gray wedge (e.g. a linear blend texture) would be hardly visible in the rendering.

4.15.1 Creating Bump Maps

You can easily create Bump Maps with Blender yourself, this is especially useful if you have modeled some small

details on a surface and you realize at the end that your scene will get too complex. You could also use the Bump Maps in a 2D application like Gimp or other, similar programs.

I will create an animated bump map in highest possible quality, it is not always necessary to make such an effort. The goal is to create a wave effect and make an image sequence of it, to be used as a bump map. The original object has 600.000 vertices, the object the map is applied to has 8 vertices.

Setting up the scene

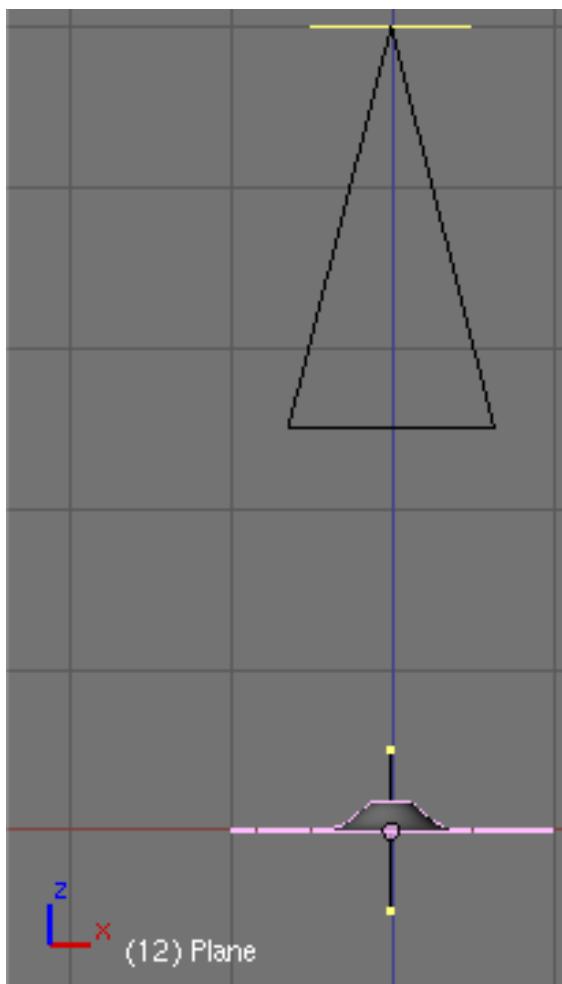


Image 2a: Setting up camera and object in the 3D view.

- Open the default scene and remove the cube.
- Insert a plane and change to edit mode.
- Crease its edges with +1 (**Shift-E**).
- *Subdivide multi* with 11 cuts.
- Change to object mode.
- Add a *SubSurf* modifier with a render level of 6.

- Add a *Wave* modifier.
 - *Time Sta:* -10
 - *Height:* 0.2
 - *Width:* 1.5
 - *Narrow:* 1.5
- Set the camera dead upon the plane.
 - *X/Y/Z:* 0/0/5. You can use the *Transform Properties Panel* to bring the camera to a certain position, the Z-Position is important in this case.
- Set the camera lens to 80.00 (*Editing Buttons->Camera Panel*).

The plane should now fit exactly into the camera view.

- Save your file.

We're going to use the Z-Buffer information to create the bump texture. The Z-Buffer contains the distance from the camera, this is exactly what a bump map is. To render the Z-Buffer information as an image, we're going to use *Composite* nodes. To get the highest possible quality, we will use *Open EXR* as file format, this allows us to store the Z-Buffer information with a numerical accuracy of 32-Bit floating point, instead of a meager 8-Bit value.

Render settings

- Change to the *Anim* tab buttons (f10).
- Set *End* to 40.
- Activate *Do Composite*.
- Select *Open EXR* in the *Format* panel, set *SizeX/Y* to both 600 (square image for the square plane).
- Set the *Output* directory to *//BumpAnim/*. This creates a subdirectory to the file where the image sequence will be stored. Don't omit the slashes.

Node editor

Now the setup for the composite nodes.

- Open a *Node Editor* window.
- Select *Composite Nodes*.
- Activate *Use Nodes*.

A *Render Layer* and a *Composite* node will be inserted automatically.

- *Add->Vector->Map Value*.

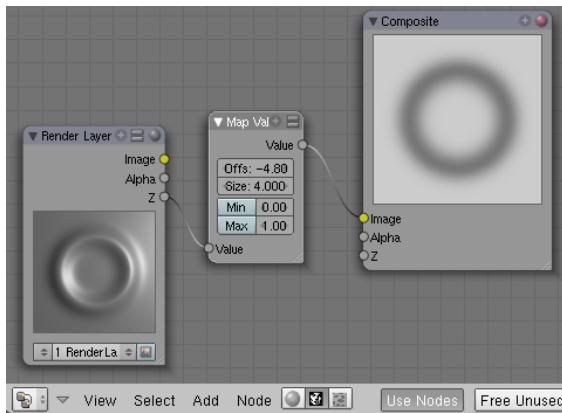


Image 2b: Composite Nodes setup to render the Z-value.

- Connect the Z-Output of the *Render Layer* to the *Value* input, and the *Value* output to the *Image* input of the *Composite* node.

If you render now the image is plain white, we have to talk a bit about the *Map Value* node. The *Offs* value is the distance from the camera where the Z-Buffer should start (in negative BU). It is not so important to get the best range, because we use OpenEXR, but if you would like to use PNG instead, you have to select this value carefully.

- Set *Offs* to -4.8 . This is the smallest distance from the camera needed. You can measure this value if you use the *Camera Clipping*, or if you render and move your mouse with pressed **LMB** above the render window.
- Set *Size* to 4 . The size value is a bit complicated. This is not directly the value range, instead $range = \frac{1}{size}$, and $size = \frac{1}{range}$.
 - If you want to include the Z-Values from -4.8 up to -5.05 the range is 0.25 , so *Size* is 4 .
 - If you want to include the Z-Values from -4.5 to -3.5 the range would be -1 , so *Size* would be also -1 .
 - If you want to include the Z-Values from -4.5 to -5.5 , the range would be 1 , so *Size* would be 1 .
- Render the animation.

4.15.2 Applying the animated bump map

- Open a new file.
- Select the default cube and change to the *Texture* buttons.
- Use the texture type *Image*.

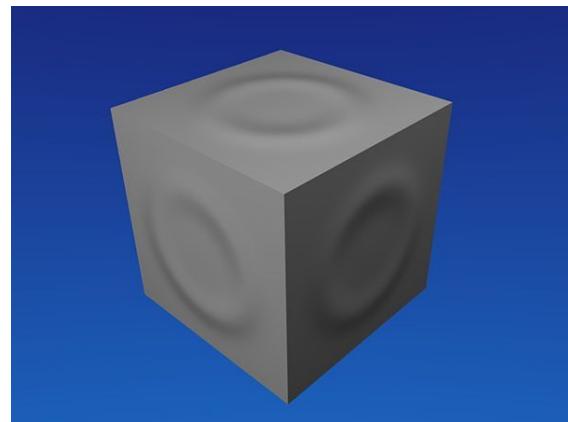


Image 3a: The Bump Map applied Video.

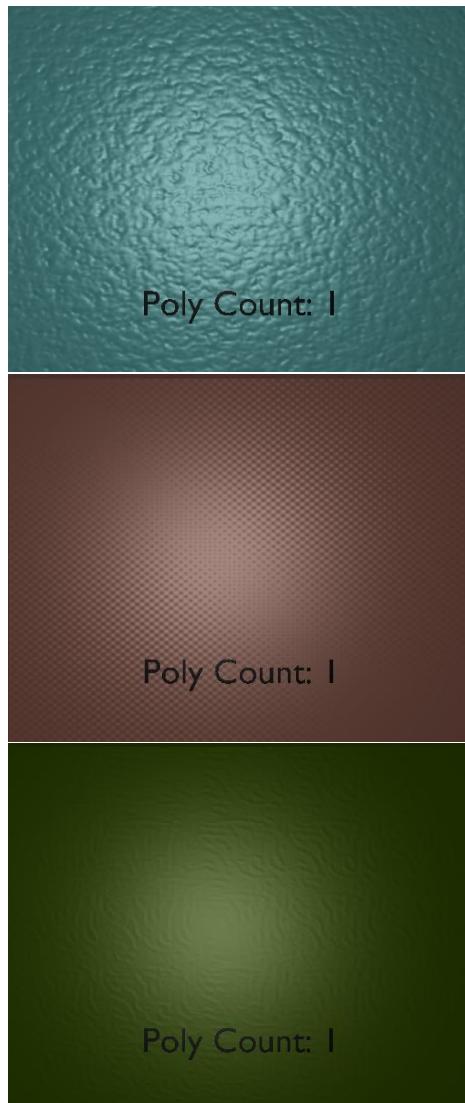
- Load the first OpenEXR-Image from the directory where you have stored the sequence.
- Activate *Sequence* in the *Image* panel, set *Frames* to 40 .
- Change to the *Material* buttons.
- Activate *Nor* for the texture and set the *Nor* slider to 25 in the *Map To* panel.
- Set the mapping to *Cube* in the *Map Input* panel.

This is it. Pretty much the same look as with the real deformation, but using much less resources.

4.16 Normal Mapping

Noob Warning: This technique is best used for models and animation, and is not well used and applied in the Blender Game Engine

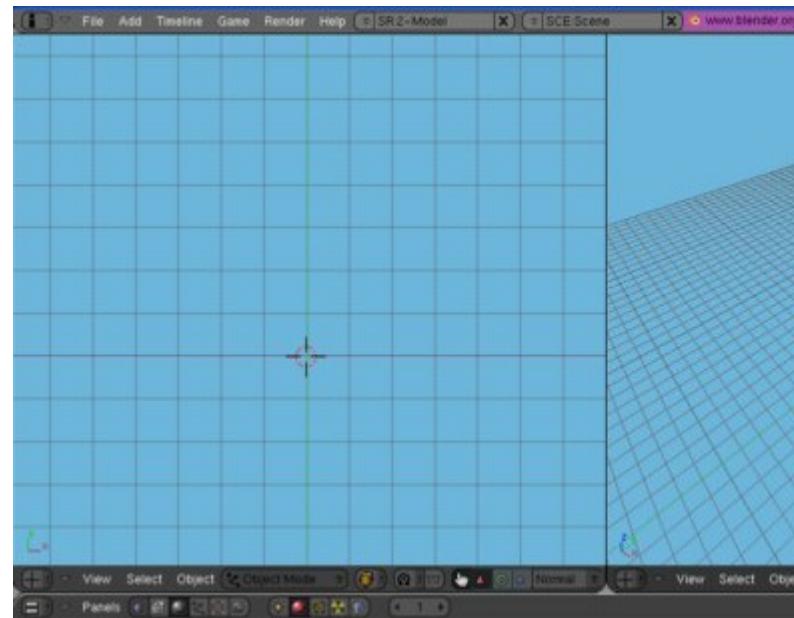
Normal Mapping in 3D graphics development is the process of using an **RGB** color-map to create a three-dimensional relief on a two-dimensional plane. The source for the normal map in blender can either be a texture already installed in Blender, or using an external picture-file (.jpg, .jpeg, .bmp, and so on) and loading it as a normal map. One great aspect of Blender and normal maps is that Blender can very easily be used to **create** normal maps that can be turned around and used in blender to reduce poly count. This tutorial will show the various ways to create a normal map and how to apply different normal maps to your model in Blender.



within Chapter:

1. Texture Normal Mapping
2. Color Map Normal Mapping

Sections



Add a mesh plane to the field, and scale that plane to five times normal size. (**Shift-AKEY**, select *Add->Mesh->Plane*, **TAB** out of edit mode, **SKEY** then **5KEY** to scale five times size.) This plane is going to be our base—it will remain untextured so it can be used as a comparison. Don't give it a material.

Now, with the 3D cursor still on the center, add a mesh cube to the field. Scale this cube 3 times in the X and Y directions, and 0.5 times in the Z direction. (**Shift-AKEY**, select *Add->Mesh->Cube*, **TAB** out of edit mode, **SKEY**, **Shift-ZKEY**, then **3KEY** to scale 3 times in X and Y directions, **SKEY**, **ZKEY**, then **.KEY-5KEY** for Z.) This cube will serve as our texture comparison.

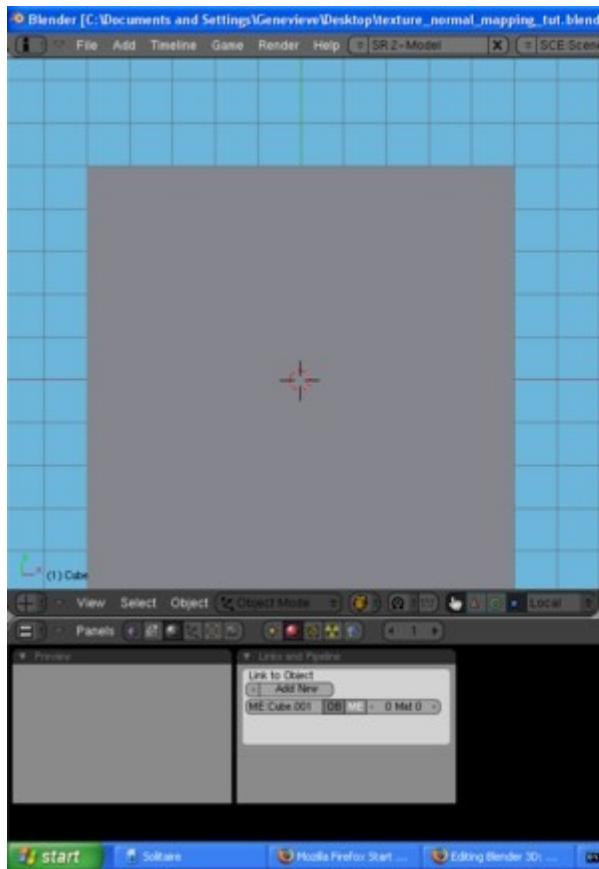
4.17 Texture Normals

4.17.1 The Texture-Normal

To familiarize yourself with normal maps and how they work, we will begin by using the texture engine in Blender to create our first normal map. This process is very useful for creating quick and non-specific normal maps for your projects that need a bit of texture.

To begin, open blender and whatever your basic load settings are, in the 3D view, delete everything so the field is clear. (**AKEY** to select all, then **XKEY** to delete)

Move the 3D cursor to the center of the X,Y, and Z axis, and switch to top view (**NUM7**) orthographic view (**NUM5**). Hit **Shift-SKEY-4KEY** then **CKEY** to center your view now.

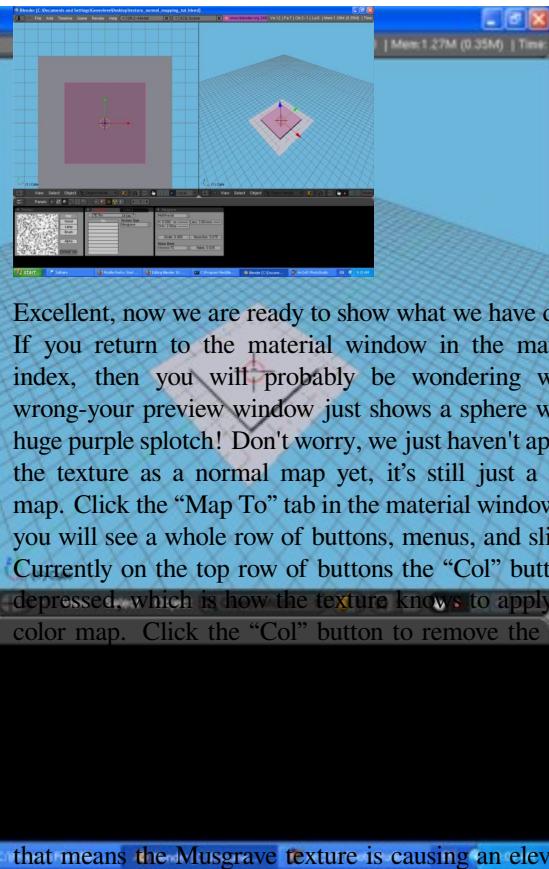


For this tutorial, let's use the "Musgrave" texture to make a 3D relief.

Select the cube, and then in the Buttons Window select the material index. Give the Cube a material, and make this material any light color you wish to differentiate it from the base plane (darker colors won't show the relief that well). Now select the texture icon in the material window. Add a new texture, and make that texture Musgrave.

Make the settings for Musgrave the following:

- Type: Multifractal
- H: 0.5
- Lacu: 3
- Octs: 2
- iScale: 0.45
- NoiseSize: 0.075
- NoiseBasis: Voronoi F2
- Nabla: 0.025



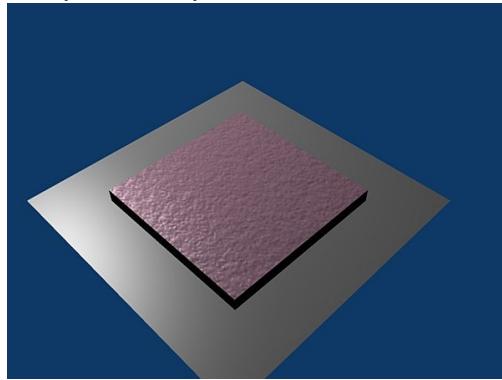
Now that means the Musgrave texture is causing an elevation texture. Click it once more and the texture is turned off. Keep the "Nor" button depressed for this tutorial.



Now that you know how that works, let's see what it did to your cube. First, go to front view (numpad-1) and zoom out a bit so about 10% of the screen is filled with your cube. Select the cube and hit Shift-S-key-4-key then C-key to center your view. In the upper-left corner, add a sun-lamp (Shift-A-key, Add->Lamp->Lamp). Adjust the "Dist" Slider to a very far length, to make sure it reaches the cube. In the upper-right corner (make it a little closer to the cube) add a Camera (Shift-A-key, Add->Camera).

Move the Camera to get a better perspective of your work, by rotating it to look at the cube and the plane. If you want exact directions: Switch to top view (numpad-7). Move -4 Units in Y direction. Rotate -75 degrees on Z rotation. Rotate -45 degrees on Y rotation. Now hit numpad-0 to see how well pointed it is to the cube. Make adjustments to your own work if you'd like.

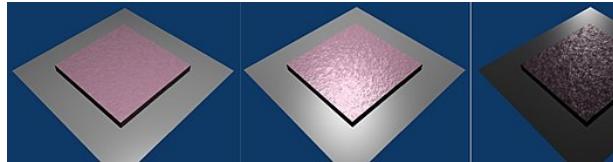
Now you are ready to Render and see how it turned out!



After your first render, I'll let you know that you can adjust the depth of the shading. Select the cube, and in the material index in the "Map To" tab, in the mid-right area, there is a slider called "Nor" that should be set at 0.5. Just to test, set it to 0 and render to see what happens. Then set it to 2 and render to see the results.

After you see it rendered, you probably realized that the edges of the cube were very sharp and didn't have the rocky texture. This is because the 3D relief on the cube is fake--it's not actual 3D, it's the computer just calculating where light should bounce off and in what direction. Imagine how many faces it would have taken to create this texture shading without Normal Mapping.

If you want, you can move the lamp around and see how the light moves around the fake-bumps.



This is best if you just need a non-specific texture, such as asphalt, cement, carpet, or even fabric. Now if you want to do something like make a low-poly brick wall, you will need to a specifically-tailored normal map for such thing. The next section will show you how you can make your own normal map and apply it as well.

4.18 Color Map Normals

4.18.1 The Color-Map-Normal

For more advanced Blender users, you probably have at one time or another wanted to create something very detailed and realistic, only to realize that one modeled head or one cool building has taken an enormous amount of memory, and rendering (or worse, animating!) something like that would probably take days! Well there is an awesome trick you can use to create intricate details and design shading, with a relatively low amount of faces. You can use blender to create and apply normal maps, which are RGB colored maps that can be used as a texture to

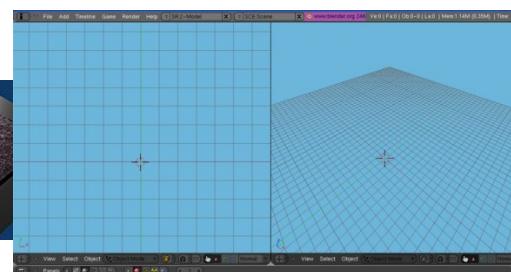
calculate how light will bounce off an objects surface. All you need is blender and photo-editing software to tweak the picture a bit.

4.18.2 Creating a Normal Map

You can very easily create a normal map once you figure out all the necessary settings. You can start by either making a high-poly count version of your object, or making the material that will be applied to the object to make the normal map.

High-Poly Version

We will first start by making a simple object with a high-poly count. Start up blender, and let's delete everything there first, then go to top-view, orthographic, and start by adding a camera.



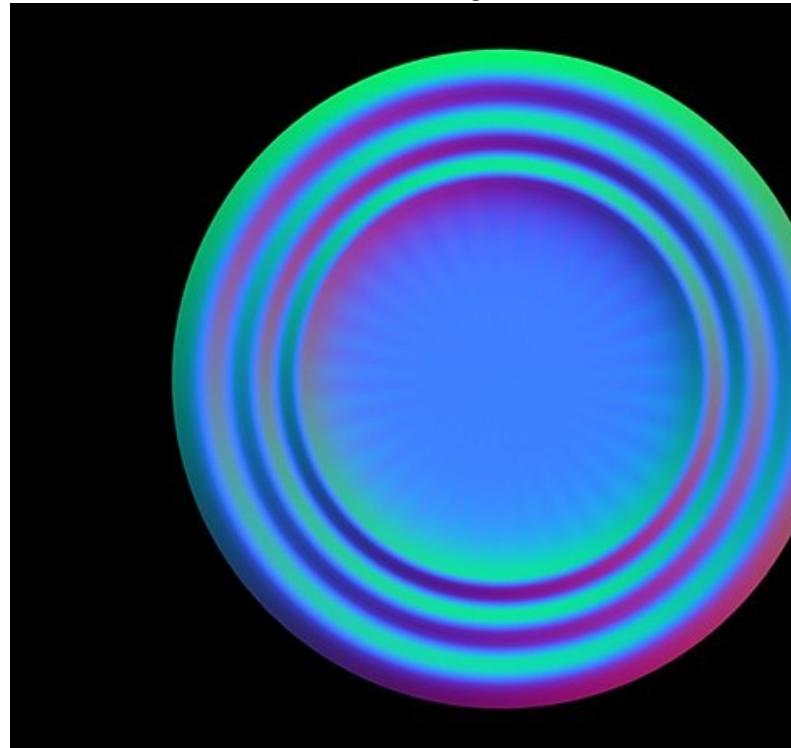
Next, start with a circle, with a radius of 2 and 32 vertices in circumference. Keep it filled. Select the edges of the circle, keeping the inner vertex unselected, and extrude edges only. Scale the new vertices by 1.1. Repeat this with only the outermost vertices 6 more times, until you have a ring of 7 vertices and a center vertex. Now, we will refer to the center vertex as center, the innermost ring as Ring 1, and the outermost ring as Ring 7. Deselect all vertices, then select all vertices of ring 2, 4, and 6 (This can easily be done by holding the Alt and Shift keys, then selecting an edge adjoining two vertices of the same ring for each ring). With these rings selected, move it 0.4 in the Z direction. Now, let's smooth this bowl out. In the editing index, under the modifiers tab, add a subsurf modifier of 2, and then apply the modifier. Then under the links and materials tab, select Set Smooth. Next after the object looks all nice and smooth, select your Camera and move it positive in the Z direction so it's positioned directly above the object looking straight down at it, and edit the camera Lens properties so that it is set to Orthographic, and scale the lens so that it includes all the bowl and fills up the camera space as much as possible.



Creating the Color-Map Material

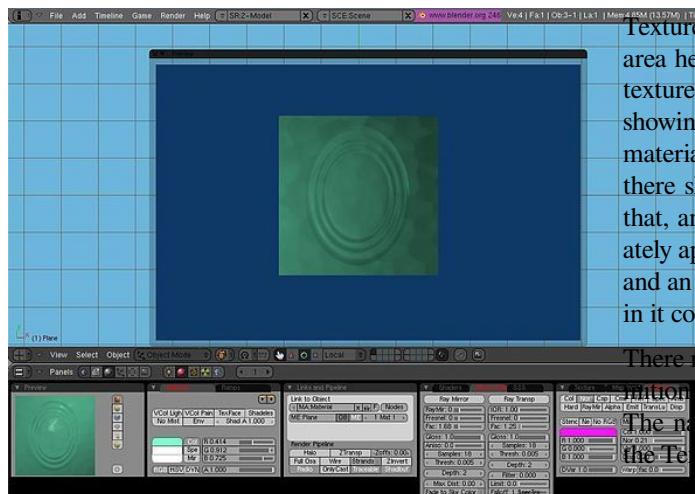
Now, to be able for the Render to apply the colors on the right areas of the object. For this, we will start by applying a material the bowl. Make this material 0.0 on the R, G, and B spectra, and depress the shadeless button. Now, go to the textures index and add 3 textures to the material, and set them as "Blend" textures with "Lin" depressed (a linear calculation). For these 3 textures, name them something to do with either "Red"/"Green"/"Blue" or "X"/"Y"/"Z" in that order, because doing this will ensure you keep the channels in order. The following image will show all settings that need to be set to make the material work.

it won't interfere when you apply it as a normal map. Now Render your image, and hit F3 to save. It should come out to something like this.



Now before your Rendering you want to make the World texture to be completely black, so this way

by adjusting the **Vol** slider directly below the **Mix** drop-down menu. I used about 0.2 and it came out just fine, like this picture.



**Congratulations, that's all there is to normal mapping.
It's a simple technique with many great uses!**

4.19 Introduction

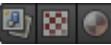
Blender's *Node Editor* lets you assemble various processing blocks (*nodes*) into combinations which feed data to one another along connections that you specify to produce complex effects. These effects can be used in three different ways: as textures, as materials, or for compositing. The Node Editor makes different kinds of processing blocks available, depending on which of the three kinds of effects you are producing:

- Texture nodes
- Material nodes
- Compositing nodes

4.20 Texture Nodes

Texture nodes allow you to produce textures that are the result of complex computations. This tutorial will just scratch the surface of what's possible.

4.20.1 A Simple “Rainbow” Texture

Start with a new Blender document. Delete the default cube. Add a new plane object. In the Material context in the Object Properties window, click the icon to the left of the material name to assign a material to the plane (there should already be a default unused one called “Material”). In the Texture context, there should already be a texture called “Tex” assigned to this material. Now split the area showing the 3D view into two side-by-side areas. In the right-hand area, bring up the Node Editor. In its area header, you should see a group of 3 little icons next to the menu ; click on the middle one for

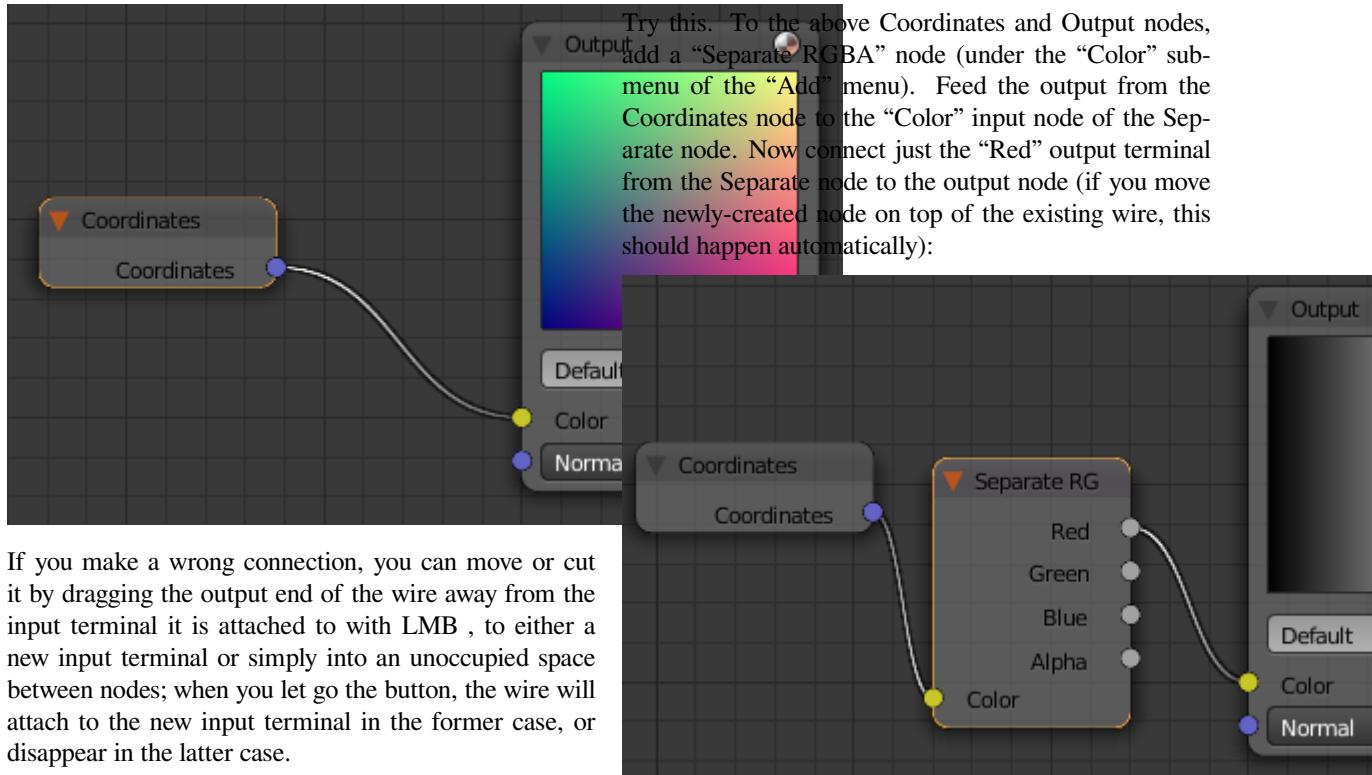
Texture Nodes. You will see appear, further along the area header, a popup menu containing the names of the textures so far created in your document: this should be showing the name “Tex” of the texture you assigned to the material for your plane object. And to the right of that, there should be a checkbox titled “Use Nodes”. Check that, and you should see a pair of initial nodes immediately appear in the editor: an *input* node titled “Checker” and an *output* node titled “Output”, with an editable field in it containing “Default”.

There must be at least one output node in the texture definition; the data fed to this will make up the final texture. The names you assign in the Name field will appear in the Texture context in the Object Properties window, in a menu titled “Output:”, with one item for each output node. Initially this may show “Not Specified”: change it to “Default”.

Each node window has little coloured circles (terminals) on its left and right edges; the ones on the left edge (if any) are inputs for feeding data from other nodes, and the ones on the right edge (if any) are for supplying data to other nodes. Thus, output nodes have no outputs (they're the final destination for the data), while input nodes have no inputs. Other node types represent intermediate stages in the processing, so they will have both inputs and outputs.

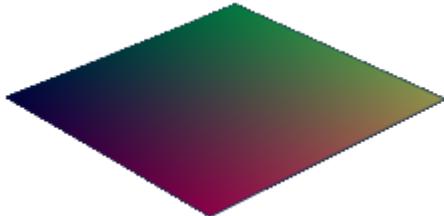
Back in the Node Editor, click on the titlebar of the “Checker” node window and use either X or DEL to delete it. Note that, unlike deleting objects in the 3D view, there is **no** confirmation popup: the node immediately disappears. (Of course, you can use CTRL + Z to undo operations in the usual way.)

Now go to the “Add” menu (or use the usual SHIFT + A shortcut), find the “Input” submenu, and select the “Coordinates” item. This will add a new input node, which just produces the unadorned texture coordinates as its data. The newly-added node will follow the mouse around; press LMB once you have moved it to a convenient place. You will see the new node has one output terminal, titled “Coordinates”. Left-click on this, and drag to the input terminal in the Output window with the word “Color” and a small rectangular colour swatch next to it. Voilà! A line (effectively a wire) should appear connecting the two terminals, and you should see the big square colour swatch in the Output window change from black to a whole rainbow of colours. That's your texture!



If you make a wrong connection, you can move or cut it by dragging the output end of the wire away from the input terminal it is attached to with LMB , to either a new input terminal or simply into an unoccupied space between nodes; when you let go the button, the wire will attach to the new input terminal in the former case, or disappear in the latter case.

At this point, you should be able to hit F12 to render, and you will see your plane object with the rainbow texture applied. You can see why I chose a Plane object: being flat, the entire texture is visible from one camera angle.



See how the Output swatch shows just a greyscale ramp running from black on the left to white on the right? This shows how the first dimension of a vector is interpreted as the red component in an RGB colour, but as the X coordinate in a position. Remove the output from the Red terminal (delete the existing wire), and take it from the Green terminal instead (by dragging a new wire from Green to the Color terminal on the Output node); now you will see the ramp going from black at the bottom to white at the top. The second dimension of a vector is the green component in an RGB colour, and the Y coordinate in a position.

The terminals have different colours to remind you what type of data they *should* be used for (purplish for coordinates, yellow for colours, grey for scalars), but as far as **Blender's Node Editor is concerned, colours and positions are not actually different types**. Both are vectors, and one can be interpreted as the other. This can be (ab)used for some creative effects! For example, the name of the “Separate RGBA” node says it’s for separating an RGBA colour into its components, but here we are using it to take apart a coordinate vector instead.

Note also in the above, that the Color terminal on the Output node expects a vector RGB colour, but when you feed it a single colour component (i.e. a scalar), it simply replicates it the necessary number of times to make a vector of the required dimension. Since all components of the RGB colour are the same, you get a shade of grey.

4.20.2 Scalars Versus Vectors

The wires carry numbers between nodes. These numbers can be of two kinds: a *scalar* is a single real quantity, like “0.5”, while a *vector* can consist of two, three or four scalars in a sequence, like “(0.5, 0.75, 0.2, 1.0)”. The number of components in a vector is also known as the *dimension* of the vector. Two-dimensional vectors can be used to represent texture coordinates, while three-dimensional vectors can represent positions in space. Colours can be represented with 3 dimensions (R, G,B or H, S, V) or 4 dimensions (RGB or HSV plus alpha channel).

Some nodes operate on scalars, while others operate on vectors. And there are nodes where some terminals input or output scalars, while others input or output vectors.

4.21 Material Nodes

Material nodes look very similar to texture nodes, and indeed there is a fair amount of overlap in functionality. Some important differences are:

- Functions for performing spatial transformations (e.g. rotation and scaling) are only available for texture nodes.
- Functions for altering diffuse versus specular colours and rendering parameters, like translucency, are only available for material nodes.

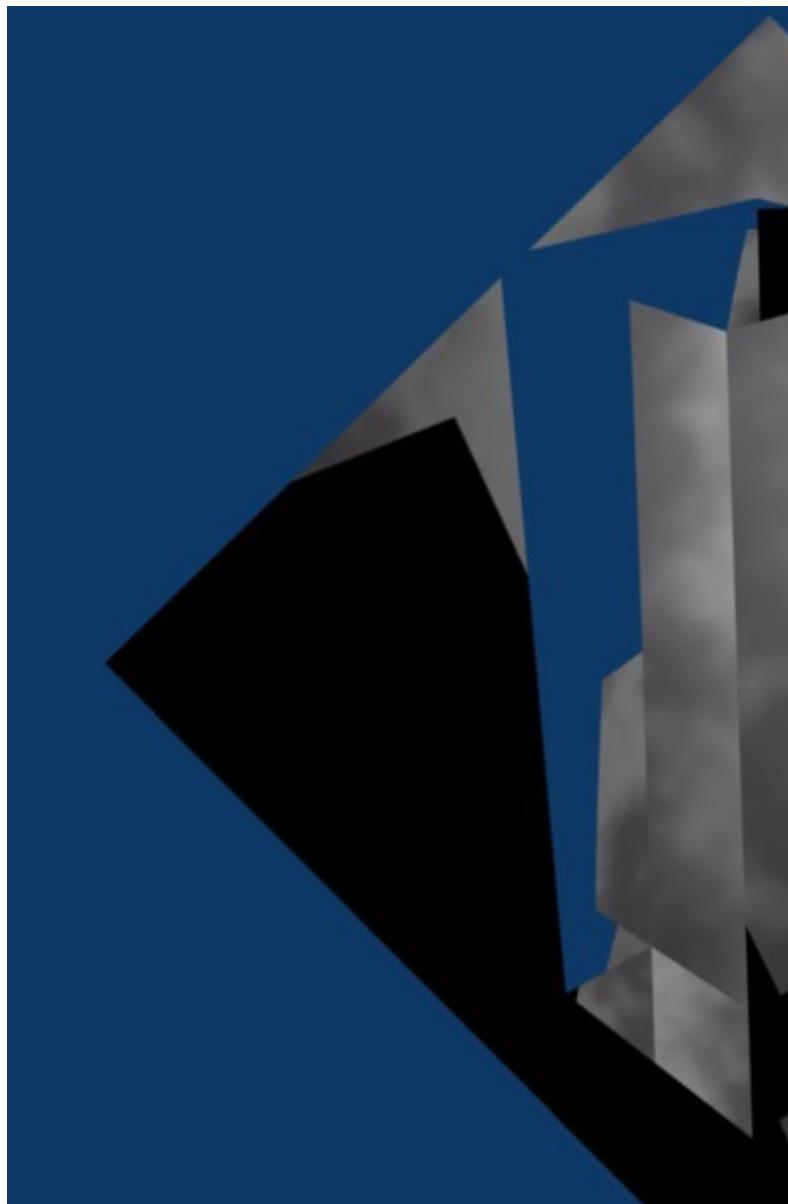
4.21.1 A Simple Graduated Material

First of all, let's set up a really basic modelling scene to show off the node-based material you'll be creating in a moment.

Open a new Blender document, with the default cube, default light and default camera. Create a plane object. Rotate and position that so it lies behind the cube from the camera's viewpoint. Scale the plane up to say, 4x, to form a decent-sized backdrop. Assign it a new default material, and a texture with some detail to it; I used a simple Marble with default settings, except under the "Map To" miniwindow I changed the colour to a medium grey from the default magenta, to stop it hurting my eyes.

Now select the cube. Scale it 3x in the Z-axis; the elongated proportions should show off a gradation more nicely. In the "Material" miniwindow, set the "Col" (diffuse) colour to white, and the alpha to zero. In the "Mirror Trans" miniwindow, turn on "Ray Transp" and set the "IOR" to some suitable refractive index, say 1.5 for glass.

Hit **F12** to do an initial render, just to confirm you've got everything nicely arranged.



With the cube still selected, open the Node Editor. By default, the Material Nodes mode should already be selected, and the combo box should be showing the name of the material you assigned to the cube; select it if it's not. Click the "Use Nodes" button, and immediately two node windows should appear, a Material one and Output one.

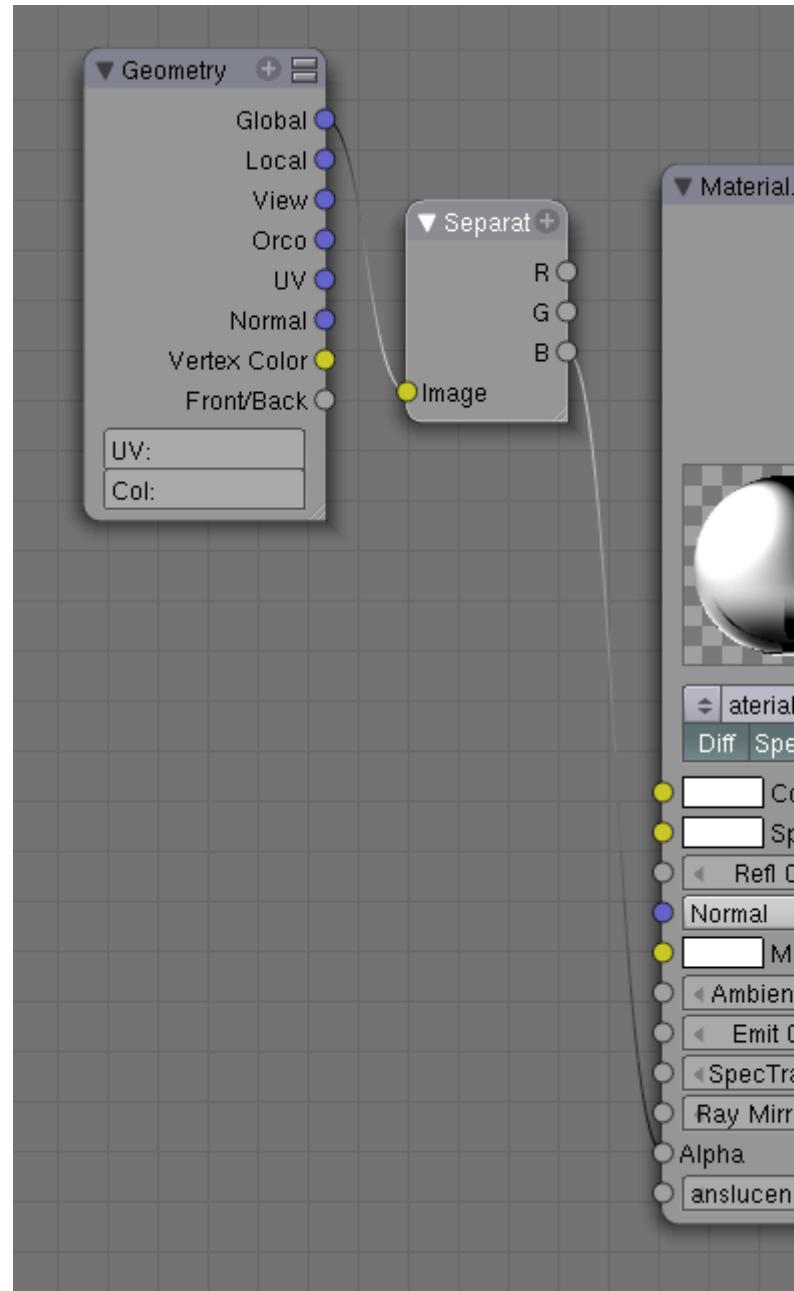
Click on the title of the Material node window, and use **XKEY** or **DEL** to get rid of it. Go to the "Add" menu, "Input" submenu, and add a new "Geometry" node. Also add a new "Extended Material" node. And from the "Converter" submenu, a new "Separate RGB" node. In the middle of the Extended Material node window, there is a popup menu showing you the names of the materials in your document; select name of the material you previously assigned to the cube again.

Make sure there are no wires running between any of the nodes; now run one from the "Color" output terminal on the Material node to the "Color" input on the Output

node, and one from the “Alpha” output on the former to the “Alpha” input on the latter.

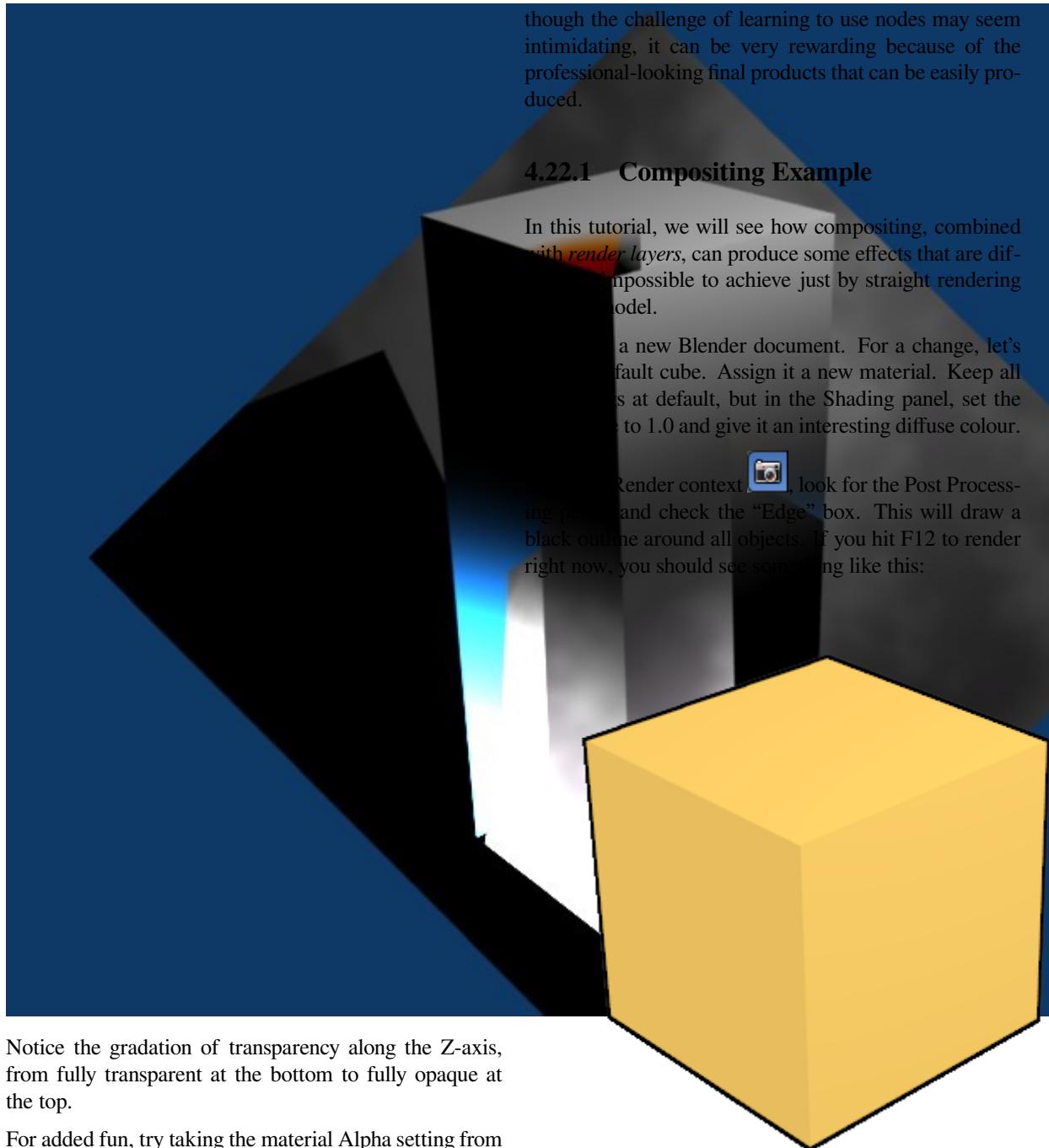
Notice that, next to each of the input terminals in the Material window, there is either a field for entering a scalar value (grey terminals), or a button which pops up a little set of controls for letting you specify X, Y and Z components for a vector (blue terminals), or a colour swatch which pops up to let you pick a new colour (yellow terminals). This way, you don’t have to make connections to all the inputs, or indeed any of them; any unconnected inputs will be set to the constant value that you specify.

Anyway, connect a wire from the “Global” output of the Geometry node to the “Image” input of the Separate RGB node. “But geometry isn’t a colour!?” I hear you cry. But as explained on [Texture Nodes](#), a vector is a vector. Feeding in a position instead of a colour to Separate RGB means its outputs are not the R, G and B components of the colour, but the X, Y and Z components of the position. So run another wire from the B (i.e. Z) output of Separate RGB to the Alpha input of the Extended Material window. The resulting layout should look like this:



Now you’re ready to render. But before hitting **F12**, hit **JKEY** to switch render buffers. That way you still have the previous render to compare with. Once the render window comes up, you can hit **JKEY** to alternate between the two renders.

Here’s what my node example looked like (note I turned the energy of the lamp down to 0.5, otherwise it seemed too bright):



Notice the gradation of transparency along the Z-axis, from fully transparent at the bottom to fully opaque at the top.

For added fun, try taking the material Alpha setting from the R or G outputs of the Separate RGB node instead, just to see the gradation orient along the X or Y axes.

4.22 Compositing

Compositing is the combination of multiple sources of visual input into a single, final image. This is common functionality in expensive, commercial video editors. Some of these (such as Adobe After Effects) use a layer-based compositing engine. Others (such as Apple Shake, The Foundry's Nuke, and eyeon fusion) use a more powerful node-based compositing engine. Blender features a very advanced and powerful compositing engine through the use of the Nodes Editor window. Al-

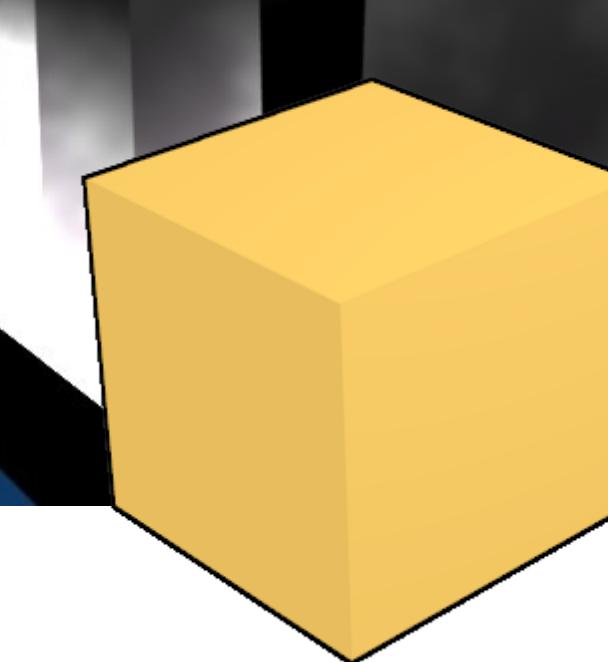
though the challenge of learning to use nodes may seem intimidating, it can be very rewarding because of the professional-looking final products that can be easily produced.

4.22.1 Compositing Example

In this tutorial, we will see how compositing, combined with *render layers*, can produce some effects that are difficult or impossible to achieve just by straight rendering a model.

Open a new Blender document. For a change, let's start with a fault cube. Assign it a new material. Keep all settings at default, but in the Shading panel, set the Alpha to 1.0 and give it an interesting diffuse colour.

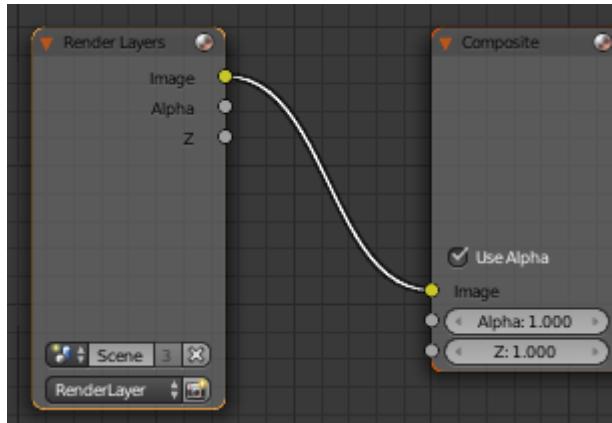
In the Render context , look for the Post Processing panel and check the “Edge” box. This will draw a black outline around all objects. If you hit F12 to render right now, you should see something like this:



So far, not very exciting. Now go to the Render Layers context . You should see there is already an initial default layer called “RenderLayer”. If you look in its Layer panel, you should see checkboxes indicating all the passes of the rendering process to include in this layer. Look for the “Edge” checkbox, and uncheck that. Now click the + icon next to the list of layers, to add a new layer. Look in its Layer panel, make sure “Edge” is checked, and uncheck all the other checkboxes in the “Include:” section.

Now open up a Node Editor window, and look for the three icons to the right of the menus, showing what kind of nodes to edit: ; click on the middle one

(if it's not already selected) to enable compositing nodes. Now check the "Use Nodes" box to the right of it. You should now see the initial default node setup, with a single Render Layers node feeding the initial default render layer directly to the output:

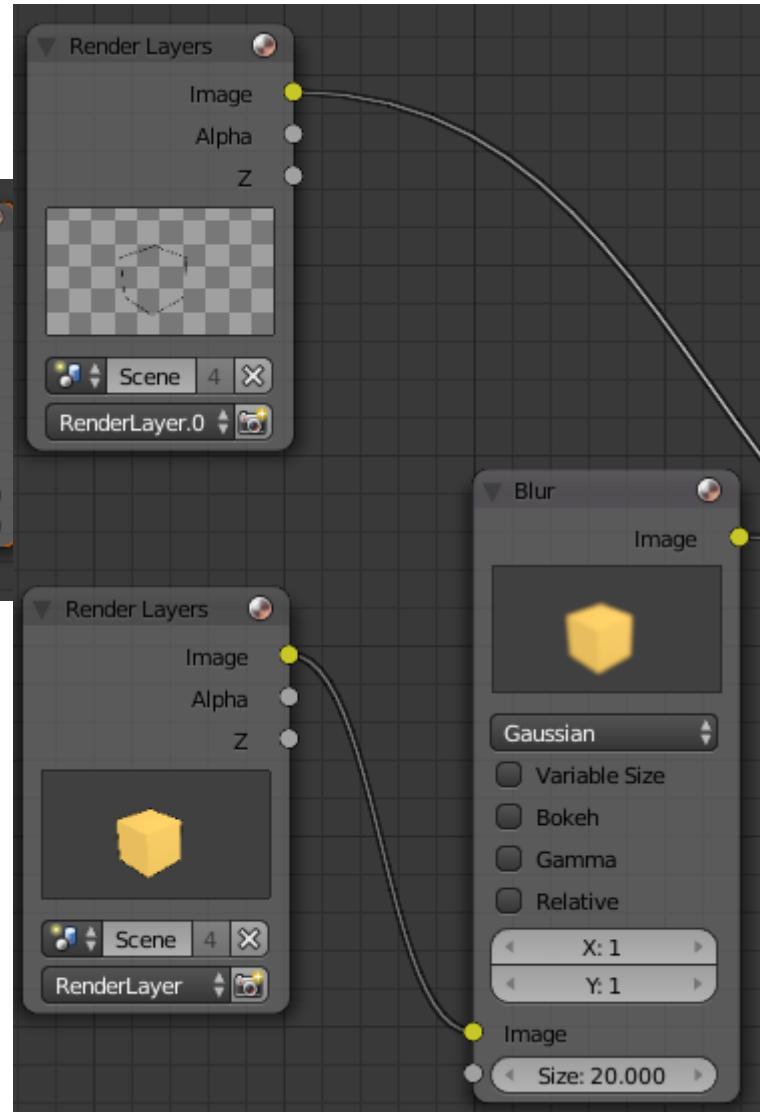


Add another Render Layers node. From its layer popup menu, select the second render layer you created, the one producing the edge outlines (which should be called "RenderLayer.001" if you didn't change its name).

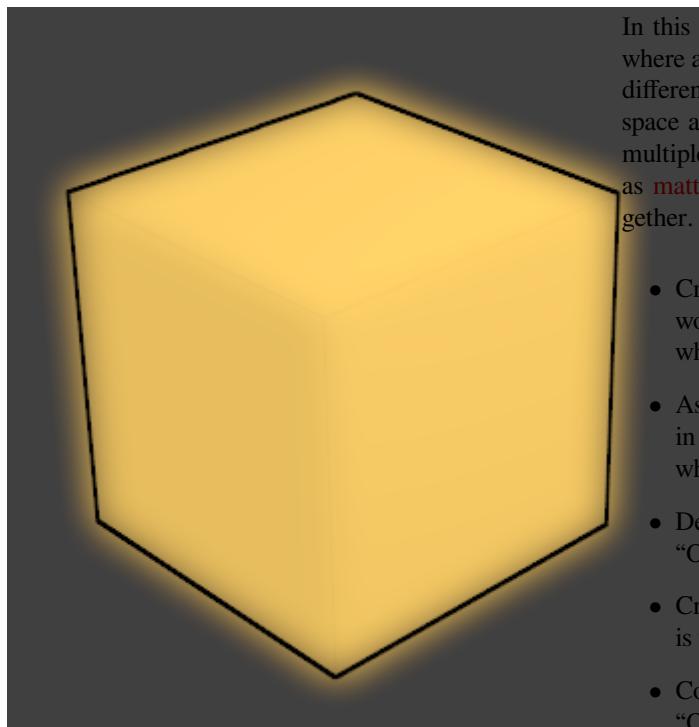
Now, add a Blur node (Filter submenu of the Add menu). The default Gaussian kind of blur will do fine. Put nonzero values (e.g. 1) into the X and Y fields (otherwise the blur Size value will have no effect). Give it a Size value of, say, 20. Connect the Image output of the first render layer (the one doing everything except the edge outlines) into its Image input.

Add an Alpha Over node (from the Color submenu of the Add menu). Feed the Image output from the Blur node into its *upper* image input, and the Image output from the second render layer (the one producing the outline edges) into its *lower* image input. Leave the Factor at its default 1.0. Connect the Image output of this Alpha Over node to the Image input of the Composite node (the final output).

When you are done, check that your connections look like this (remember, the exact positions of the nodes are unimportant, except for clarity; what's important is the connections between them):



The effect of this is to blur the rendering of the cube (apart from the edge outline), and then put the edge outline back on top of it. If you hit F12 to render now, you should see something like this:



In this tutorial, we will see how to create an animation where a doorway leads from one world into a completely different world that clearly cannot be occupying the same space as the original world. We will be making use of multiple *scenes* within a single Blender document, as well as *matte nodes* in the compositor to combine them together. The overall workflow will consist of these steps:

- Create a scene representing the “Outer” world (the world outside the doorway, which is also the world where the observer is located.)
- Assign a specially-coloured material to the doorway in the “Outer” scene, which will become transparent when composited.
- Define a camera movement panning around the “Outer” world, to accentuate the portal illusion.
- Create a scene representing the “Inner” world, which is visible through the doorway.
- Copy the exact same camera movement from the “Outer” world to the “Inner” one, so the view through the door remains consistent with what is seen on both sides. (This is why it is best to keep the two scenes in the same Blender document.)
- Render out separate animation sequences for the “Outer” and “Inner” scenes.
- Create a third scene purely for compositing purposes, bringing in the two previously-rendered animation sequences.
- Render out the complete composited result sequence.

4.22.2 Things to Try

What happens if you put different values into the X and Y fields of the Blur node? Try making the X value different from the Y value.

What happens if you get the inputs the wrong way round into the Alpha Over node?

Try the other blur types available from the menu in the Blur node.

4.22.3 Other Tutorials

Composite Nodes Video Tutorial (2.49b):
<http://www.youtube.com/watch?v=AAx0JsdAAM>

Vector Blur Tutorial (2.49b): <http://www.youtube.com/watch?v=qY4WcNgExv8>

Depth of Field (DOF) Using Composite Nodes (2.49b):
<http://www.youtube.com/watch?v=HBomEv-bEtw>

4.23 Further Compositing: A Portal Effect

You may be familiar with the time-travelling machine known as the **TARDIS**, which is much bigger inside than it appears from the outside. Alternatively, you may have come across the concept of a *portal* in science fiction, which lets you instantaneously move from one space/time location to another, just by taking a step.

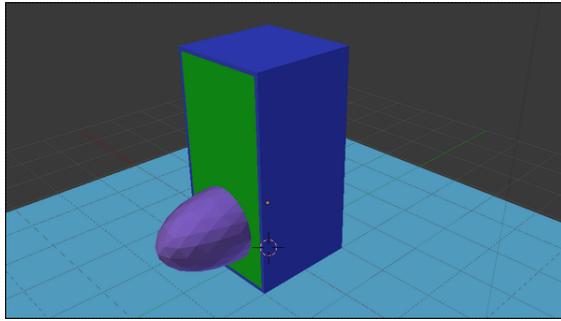
The actual modelling we are doing here will be very simple, not to say crude, just to illustrate the basic principle; feel free to create your own more elaborate and more interesting scenes, once you have mastered the basic technique!

4.23.1 The Outer Scene

Start with a new Blender document. Change the Scene name to “Outer”. We will use the default cube as our portal object (why not?). Make it taller, so the front face has about the right proportions for a doorway. Add a plane for a floor just underneath. Go back to the portal, TAB into Edit mode, select the front face, and inset it just a little bit, for a door frame. Switch to face-select mode, and select just the inner face; give this a new bright green material, and be sure to select the “Shadeless” option in

the Shading panel under Materials  settings. Making it shadeless ensures that the whole doorway has a uniform colour regardless of lighting; the exact colour doesn’t matter, so long as nothing else in the scene comes too close

to the same colour. Green is the colour traditionally used for chroma-key work in movies.



Your basic scene setup should look something like this. Apart from the doorway material as discussed above, none of the other material assignments really matter. Note the egg-shaped object (stretched-out sphere) I added poking halfway through the doorway: this object will also appear in the inner scene, reinforcing the idea of a doorway between worlds that you can actually move through.

4.23.2 Animating The Camera

Still in the Outer scene, add an Empty object at the default 3D cursor location. Parent the camera to this; now I insert a LocRot keyframe for the Empty at frame 1, then go to, say frame 51, rotate the Empty about the Z axis by -90° (so the view moves across to the opposite side of the but still showing the doorway), and I insert another LocRot keyframe.

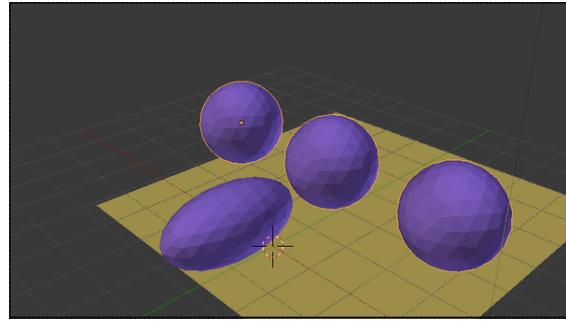
While you're at it, make the following changes in the Render Context:

- Set the end frame for the animation frame range to 51.
- In the Shading panel, make sure the output format is PNG, and set the Output path to //output frames/. Putting the rendered frames from this scene into their own directory will avoid any mixups with the other scene renders.

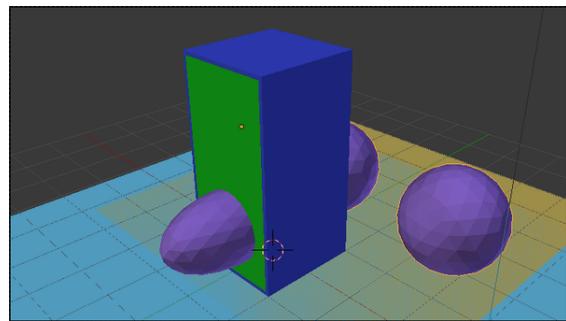
If you were to render this scene animation out now, you should end up with something that looks like at right.

4.23.3 The Inner Scene

In the Scene popup menu, click the “+” icon to create a second scene. Choose the option “Link Objects”: this means the new scene will initially share exactly the same objects as the first scene. (Sharing the same objects, rather than making copies of them, just makes it easier if you need to make any changes to those objects; those



changes are automatically visible in all scenes sharing the same objects.) Name the new scene “Inner”. We only want it to share the camera (and its pivot Empty) and lighting, and that purple egg thingy that is travelling between the worlds. So select the portal box and floor, and delete them. Create some new objects to make up this inner world, something like at right. Note I gave the inner world its own floor, with a contrasting colour.



Inner scene with Outer as background

Tip: In the Scene Context, there is a “Background” field in the Scene panel, where you can choose another scene to overlay this one. This is what I did in the screenshot at right, just so I could make sure those balls in the Inner world lie behind the portal door. Don’t forget to **remove this Background scene setting** when you are done. Otherwise the Outer scene will show up in the render for the Inner scene, which we don’t want!

4.23.4 Inner Scene Animation

When you created the Inner scene, it would have inherited the same Render settings as for the first scene: the same frame rate and number of frames to render. Just make one change: in the Output panel, set the path to //inner frames/, so as to keep the rendered frames separate from the Outer scene.

Now when you render out the animation of this scene, you should end up with something like at right.

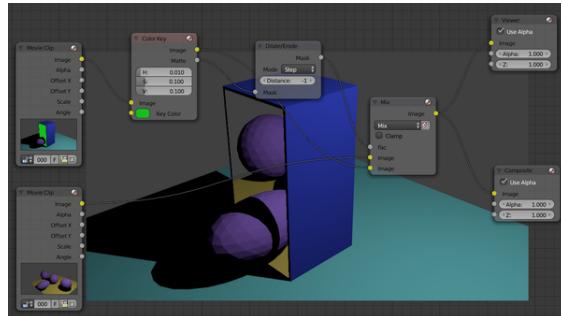
4.23.5 Compositing It All Together

Now, make a third Scene, just for the compositing. In the menu that appears when you click “+” next to the scene name, choose “New” to create a new Scene with nothing in it. Call this scene “Compositing”. Leave it empty; we won’t actually be rendering any 3D content in this scene.

Open up a Node Editor window, and change the node editor type to Compositing nodes. Check the “Use Nodes” box; you will see the usual initial two nodes, “Render Layers” (representing the rendering of the 3D objects in the scene), and “Composite” for the final output from the compositor. Get rid of the Render Layers node; we won’t need it at all (and anyway, it won’t work without a camera in this Scene).

Instead, SHIFT + A add an Input→Movie Clip node. Click its Open button, go to the “inner frames” subdirectory where the Inner scene has put its rendered frames, and select All the PNG files you find there to make up this movie clip. Click the “Open Clip” button to confirm and return to the Node Editor.

SHIFT + A add a second Input→Movie Clip node. Click its Open button, and this time go the “outer frames” subdirectory where the Outer scene has put its rendered frames, and select All the PNG files you find there to make up this movie clip. Click the “Open Clip” button to confirm and return to the Node Editor.



Now put together the complete node setup as at right. Note the use of a Matte→Color Key (*not* Chroma Key) node to make the doorway transparent. To choose the right colour, bring up the colour picker for the “Key Color” input terminal, and use the eye dropper to click on the door part of the thumbnail image in the Movie Clip node for the Outer scene. Note I had to add a Filter→Dilate/Erode node with a Distance of –1, otherwise I had a little trace of green showing at the edge of the doorway. The computed colour-key mask then feeds a Color→Mix node which causes the Inner scene to show through exactly where the door is located in the Outer scene, but nowhere else! Simple, isn’t it?

Now, make one final change to the Render settings, setting the path in the Output panel to //composite frames/ to keep them separate from the two input scenes.

Now when you render out the Compositing scene, you should end up with an animation something like at right.

4.23.6 More On Lighting

In this example, I gave both scenes the same lighting. But in general there is no reason for this; in fact, giving the two scenes completely different lighting (e.g. night in one, day in the other) would emphasize the jarring discontinuity in moving between them.

However, you might then have to deal with the issue of light spilling from one world to the other through the doorway. This could be faked by judicious placement of additional lights.

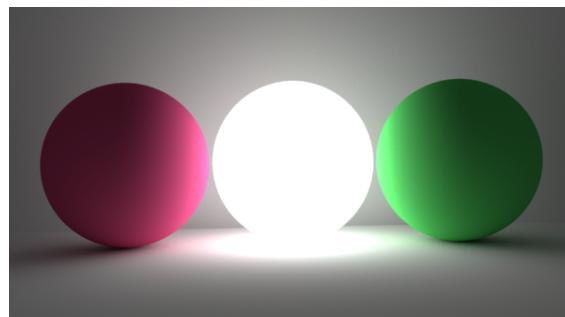
Further exploration of this issue is left up to the reader.
;)

4.24 Advanced Rendering

4.25 Alternative Renderers

You previously learned about the **Blender Internal** renderer. This is the old, original renderer that was included from the earliest days of Blender. However, it is not the only way to render scenes created with Blender.

4.25.1 Cycles



Example Cycles render showing 1) illumination via an emissive material (middle ball), and 2) colour bleed via indirect diffuse lighting (note coloured light on ground near red and green balls). The glowing middle ball is the only light source in this scene.

Beginning with Blender 2.61 came the new **Cycles** renderer. Its most important new feature is **more physically-accurate handling of lighting** in the form of **global illumination**. What this means is that Cycles automatically implements subtle indirect-lighting effects, where light can bounce off a diffuse surface and provide some illumination to other diffuse surfaces nearby. And it doesn’t distinguish in its lighting calculations between “lamp” objects and non-lamp objects which have “emissive” mate-

rials; in Blender Internal, the latter do not provide any light to other objects, whereas in Cycles they do.

Cycles also offers high-fidelity previews in the interactive 3D view. As it continues to be developed, it will perhaps replace the old Blender Internal renderer someday, but not all the features are there yet for this to happen.

See also:

- Manual on Cycles.
- Andrew “BlenderGuru” Price’s Introduction to Cycles.

4.25.2 Freestyle

This is not a complete renderer in itself, but a new addition to the Blender Internal renderer in version 2.67 of Blender. It adds sophisticated options for accentuating edges to produce cartoon-like renders.

See also:

- Manual on Freestyle.
- Report by Libre Graphics World on the origins of Freestyle.

4.25.3 External Renderers

There are also a great number of stand-alone renderer programs, both proprietary and Free Software, around (e.g. [Aqsis](#), [LuxRender](#), [Octane](#), [RenderMan](#), [Yafaray](#)). These have no capability to create or edit 3D models or scenes, they are designed purely to perform rendering on already-created models and scenes. Several of them—particularly the Free Software ones—are supported by addons to allow them to work with Blender almost as conveniently as its built-in renderers.

4.26 Disadvantages of Other Renderers

Of course, there can be a downside or two to choosing a renderer other than good old Blender Internal.

4.26.1 Say Goodbye to Your Materials and Lighting

One important thing to note when constructing models is that **materials and lighting are renderer-specific**. If you have painstakingly set up transparent materials and bump maps and all the rest of it, then decide to switch renderers, you will have to do that work all over again.

And you may find that there is no exact one-to-one correspondence between particular features of Blender Internal materials and lights and those supported by the alternative renderer, so you will have to find approximate workarounds.

Of course, the actual geometry of your models remains unaffected.

4.26.2 High-Quality Renderers Are S-L-O-W

If you are accustomed to hitting F12 and seeing an image appear after just a few seconds with Blender Internal, using one of these other renderers will likely come as a shock. To get a good-quality render can easily take **10-100 times as long** as with Blender Internal. You can speed things up by choosing lower-quality settings, but then the results tend to be filled with noise and not fit for much more than preview purposes.

4.27 Introduction to Cycles

Cycles is a renderer which first appeared in Blender 2.61. It offers more physically-accurate handling of lighting, based on a completely different handling of materials from the old Blender Internal renderer.

4.27.1 A Simple Cycles Example

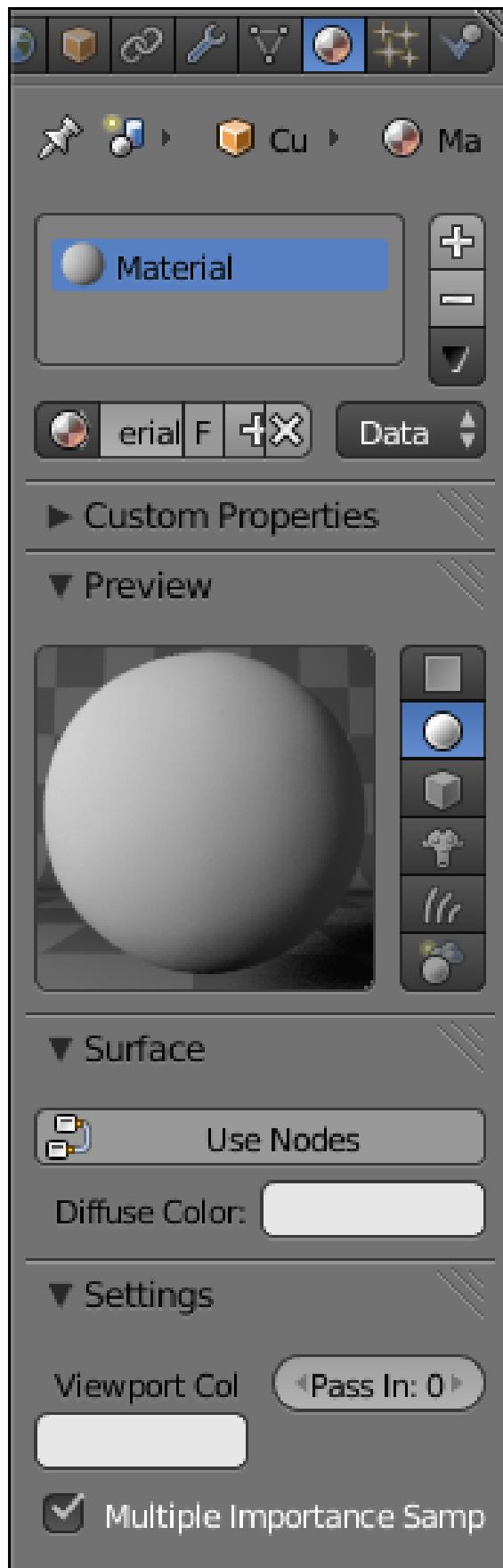
Start with a new Blender document. From the renderer



menu at the top of the screen, select “Cycles Render”. Select the default cube. Go to the Material Context in the Properties Window, and you should see something like at right.

What happened to diffuse/specular shader settings, transparency, mirror, etc?

All that is gone. Instead, Cycles defines its materials in terms of something called a **BSDF**, or *Bidirectional Scattering Distribution Function*. When light hits the surface of an object, it can be *reflected*, *absorbed* or *transmitted* (through the material, giving rise to transparency or translucency), or more typically a combination of all three. Ignoring absorption (which means the light is lost anyway), the way the light is reflected can be represented mathematically as a **BRDF**, and the way it is transmitted as a **BTDF**; Cycles unifies the two into the general concept of a BSDF.



If you look in the window at right, you will see little in the way of actual material settings except for the choice of a “Diffuse Color” for the material, and above it a “Use Nodes” button. There is a little bit more in the “Settings” panel below (including what colour to use to show the object in the 3D viewport), but we will ignore that for now.

By default, the material is assigned a simple *diffuse* BSDF (non-shiny surface), and the only parameter exposed for you to play with is the diffuse colour. To get access to more, you will need to click the “Use Nodes” button. But don’t do that yet.

Instead, *delete* the default lamp. We are going to light our scene entirely with an emissive material. Why? Because, with Cycles, we can. In real life, we can usually see the sources of light in our scene, they are not invisible the way the Blender Internal renderer treats them.

Add a new UV sphere object, and position it above the cube. Go to the Materials Context, and assign it a new material. You should see a few more settings appear this time, as at right (if you see the “Use Nodes” button as before, then click it).

Go go the menu next to the label “Surface:”. Clicking on this will show you (in the left column) the list of available BSDFs:

Select “Emission”. The “Surface” panel should now look like this:

If you hit F12 to render at this point, you will probably end up with something like this. This shows the emissive sphere shedding light on the diffuse cube, which shows Cycles in action, but is otherwise not very interesting. Also note the noise, particularly on the upper surface of the cube.

Let’s see if we can spice it up a bit.

Add a plane below the cube, and scale it out a bit to look more like a floor. You can give it the same material as the cube. While we’re at it, go back to the glowing sphere and increase the Strength of the emission material to 10.

Now if you render, you should see something like this:

OK, so that’s slightly more interesting (if you look carefully, you can see the lower part of the cube pick up some light bouncing off the floor), but that noise is even more terrible than ever. Can we do something about that?

Yes we can.

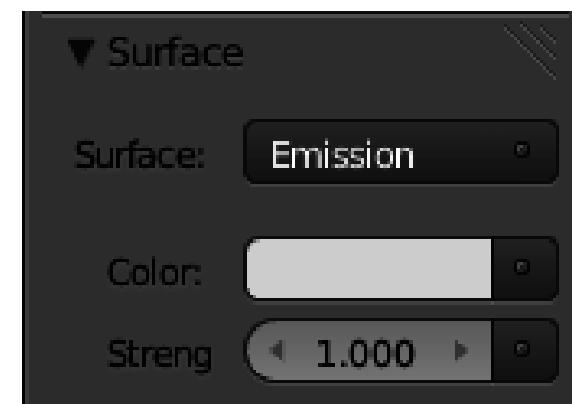
Go to the Render Context in the Properties window. Look for the Sampling panel (this is specific to the Cycles renderer, you won’t have seen it while doing renders with Blender Internal). See the two numbers in the column headed “Samples:”, in particular the one labelled “Render:”. Change the number from the default 10 to 100.

Now if you hit F12 , you should see something like at right: the noise is noticeably less.



However: note the increase in render time. On my quad-core Core i7 machine, the previous image rendered in

Shader	Link
Transparent BSDF	Remove
Glossy BSDF	Disconnect
Glass BSDF	
Diffuse BSDF	
Subsurface Scattering	
Emission	
Add Shader	
Velvet BSDF	
Translucent BSDF	
Mix Shader	
Ambient Occlusion	
Background	
Holdout	
Refraction BSDF	
Anisotropic BSDF	



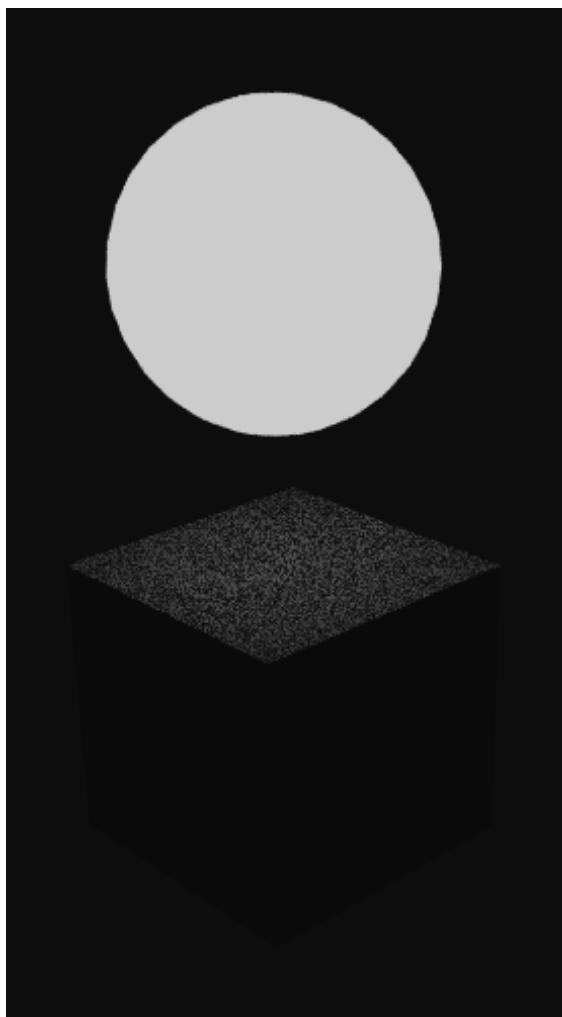
less than a second at 1500×2000 resolution, while this one took closer to 6 seconds—almost exactly a factor of 10.

You can crank up the number of render samples some more, and get an even better-quality result, but it will take even longer. This is the tradeoff with all high-quality renderers, not just Blender Cycles: you can get better results, but at the price of longer render times.

4.27.2 Real-Time Rendering Previews

Cycles has another little trick up its sleeve. In the 3D view window, look for the shading menu, where you can choose solid/wireframe/textured etc previews.

Notice anything different? With Cycles enabled, there are extra “Material” and “Rendered” items in this menu. Try selecting “Rendered”, and suddenly you will be seeing a full Cycles render happening in the 3D view! Try rotating the view, zooming in and out etc, and watch the render being redone every time: initially it starts out very



Crude render with Blender Cycles

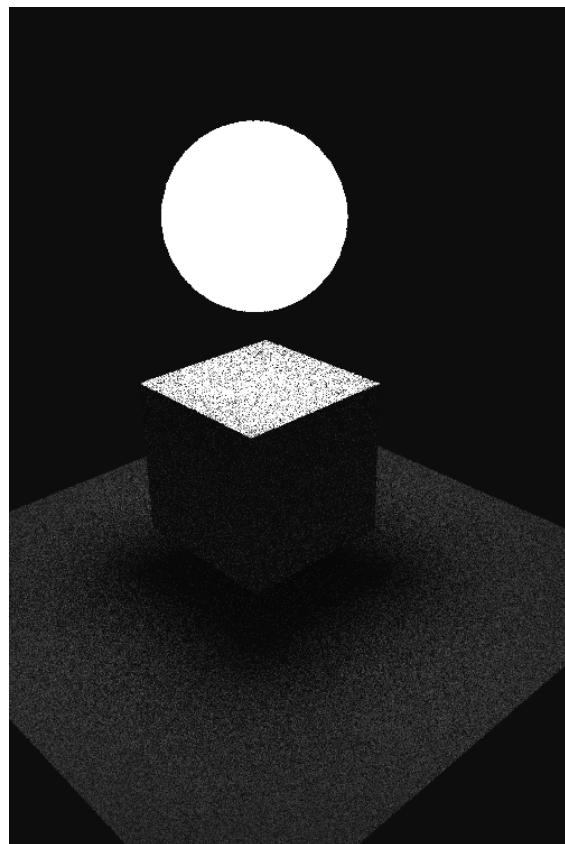
blocky and noisy, then gradually refines itself if you don't touch anything.

Remember the two render samples values we encountered above? Now you know what the lower "Preview" one is for: it controls the quality of rendering done in preview mode. You can also set this value to 0, which means "infinite": rendering becomes a never-ending process, where the quality of the image gets better and better, the longer you leave it. Of course, you can interrupt it at any time, just by doing something to the view.

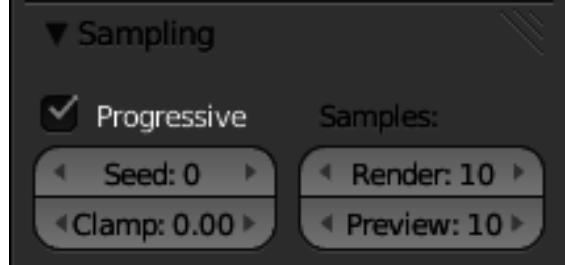
Of course, all the usual object/edit-mode functions etc remain available, but they're a bit hard to use, because there is no selection feedback in this view.

4.27.3 Nodes? What Nodes?

This example has been a very simplistic use of Cycles materials. Much more sophisticated uses are possible, by combining various BSDFs and putting them through all kinds of processing. But that requires delving into the Node Editor.

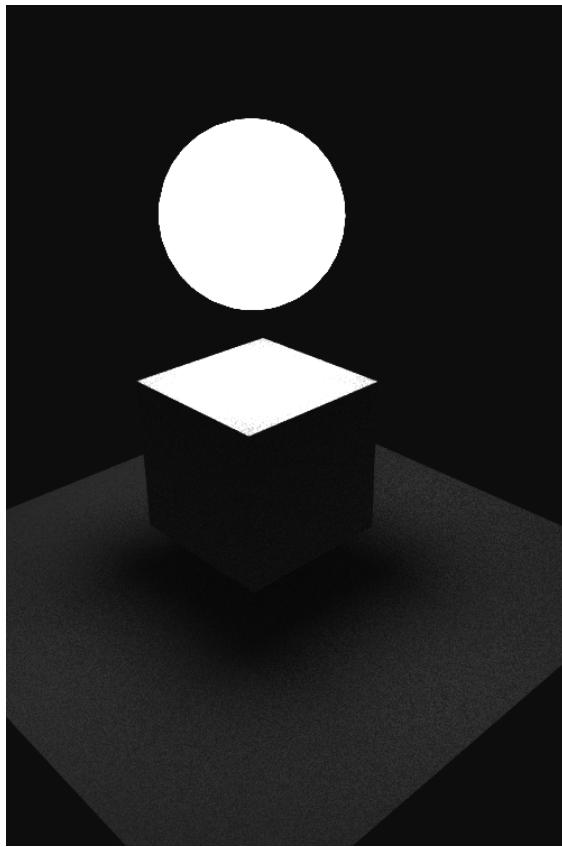


Slightly better, but still crude, render with Blender Cycles

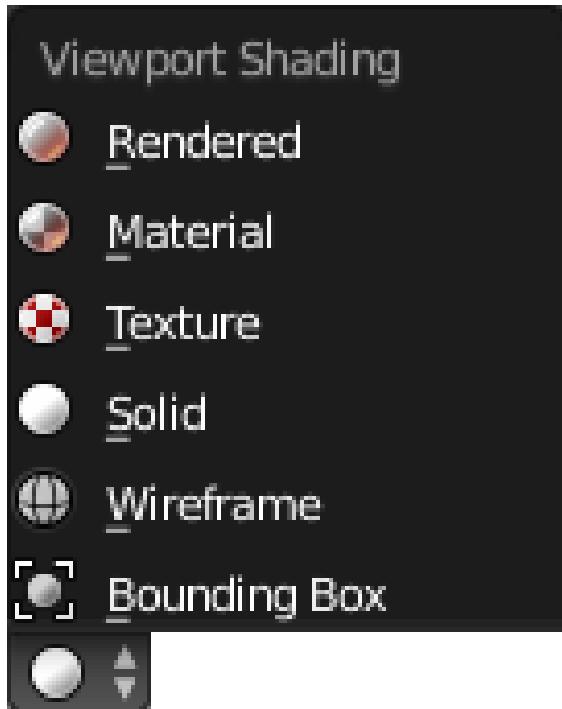


4.27.4 See Also

- Reducing noise with Cycles renders in the Blender wiki.
- Video tutorial by Peter Drakulitch showing how to use a despeckle filter to reduce "fireflies" (anomalous bright dots) in Cycles.
- Andrew Price's 4 easy ways to speed up Cycles.
- Thomas Dinges at BConf 2013 explaining various Cycles options and likely future development directions.



Getting better, crude render with Blender Cycles



4.28 A Glass Material in Cycles

This tutorial will redo the previous Ray Tracing lesson, this time using Cycles rendering. Start with a copy of

the same document you created last time; we will keep the same geometry, but redo all the materials, and then you can compare the final results to see how the different renderers perform.

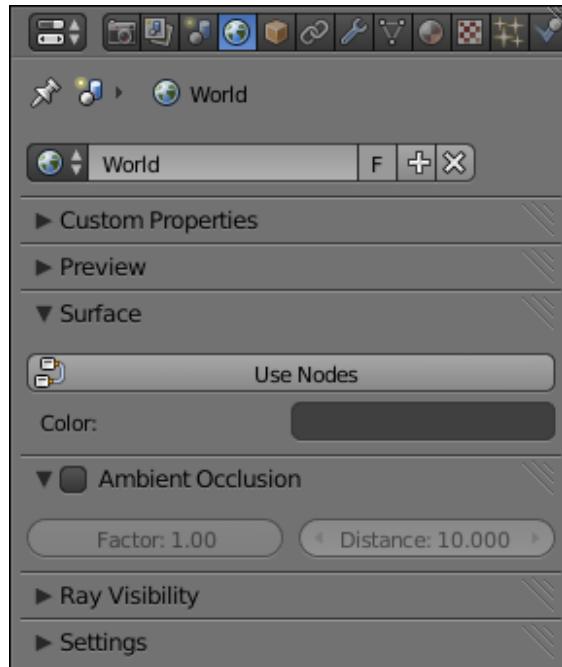
Open the ray-tracing document. Switch the renderer



to Cycles. We previously defined two materials: the world background with the “magic” texture, and the glass for the cubes. So we will need to redo these.

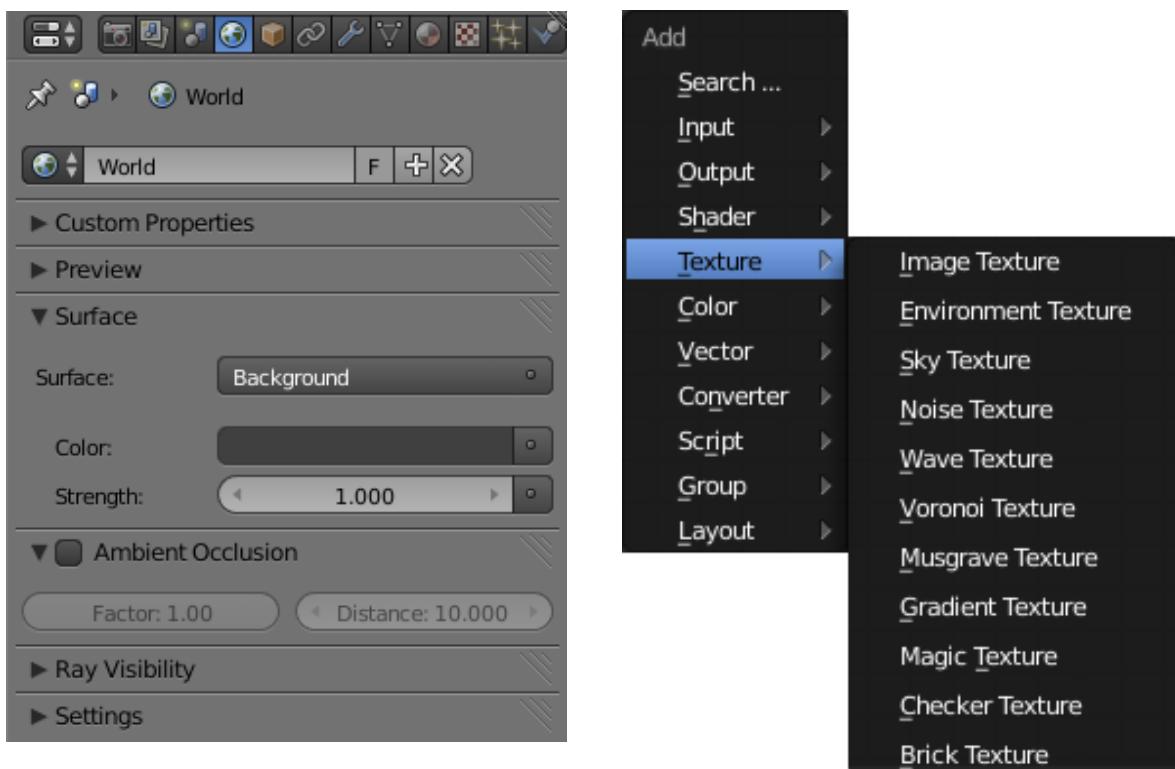
For all but the most basic plain diffuse materials, Cycles requires you to use the Node Editor. You should have done the prior introduction to this; please review that, if not. Cycles material nodes are not the same as Blender Internal material nodes; but the basic mechanics of node editing are the same.

4.28.1 The Magic Sky



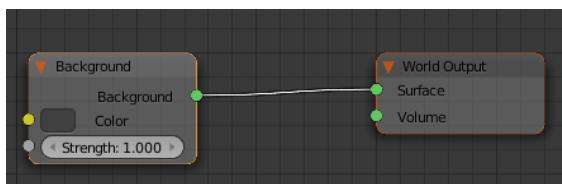
Let's try fixing the background material first. Go to the context in the Properties window. It should look something like at right. What happened to all the world settings? Nearly all of them have disappeared! But see that big “Use Nodes” button? It's big so that you will find it hard to miss. Click it.

OK! Now there are a few more settings, but still not much. For all the fun stuff, we must go into the Node Editor.



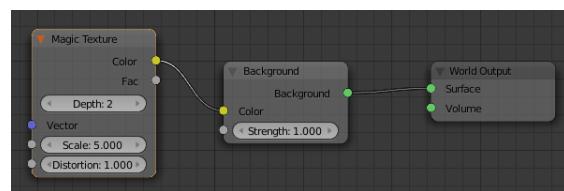
You probably have a Timeline window across the bottom of your Blender screen layout. Change the window type to Node Editor , and increase its height. (Or add a new window if you don't already have one, and make it a Node Editor.) In its header, next to the menu titles, you should see a group of 3 icons indicating what type of nodes you want to edit: if you hover over them, from left to right, you should see tooltips identifying them as respectively "shader nodes" (i.e. Cycles materials), "compositing nodes" and "texture nodes". It is the "shader nodes" we want.

With shader nodes selected, you will see another pair of icons appear immediately to the right of these three: these indicate whether to edit shader nodes for a selected Object, or for the World. It is the World that we want.

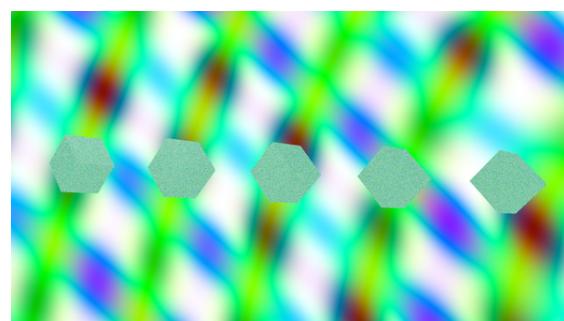


Soon as you select the World shader, the following should appear in the Node Editor window. Note the special "Background" and "World Output" nodes: you must have these in the World shader, but note that "Color" input terminal on the Background node? That's just itching to have something plugged into it.

Bring up the menu to add a node, with SHIFT + A . Look for the Texture submenu, which should look like at right. Select the "Magic Texture" item.



This will add a new node representing the texture; connect its "Color" output to the "Color" input on the Background node, like at right.

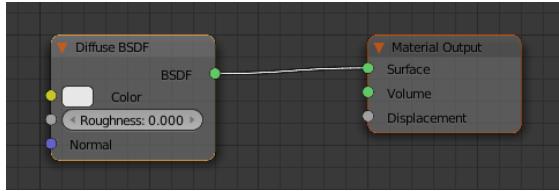


Now if you hit F12 to render, you should see your new magic World background.

4.28.2 Here Comes The Glass

Make sure the cubes are selected in the 3D view. Go to the Materials  context in the Properties window. As before, you will see very few settings, and a big “Use Nodes” button. Click it.

In the Node Editor, remember there are two icons for selecting what type of shader/material nodes to edit, the Object one and the World one? Click the Object one this time.



As before, you will see an initial default node setup, like at right.

This time, there is only one node that needs to be present for the node setup to work correctly, and that is the Material Output node. Click with RMB on the node titled “Diffuse BSDF”, and press DEL to get rid of it.

Now bring up the Add SHIFT + A menu, and this time look for the Shader submenu. We will need three items from here, so add them one at a time:

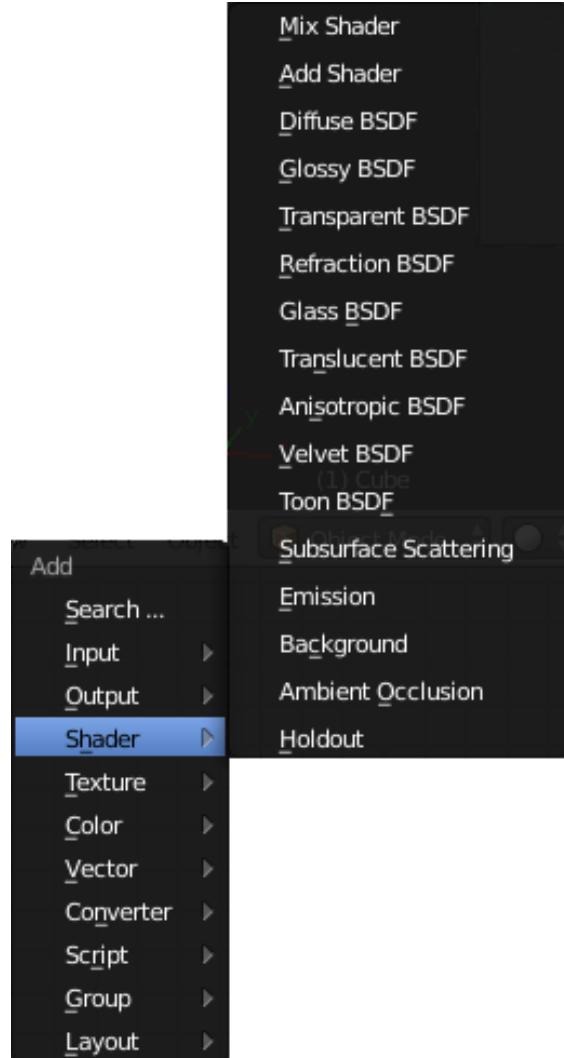
- A Glossy BSDF to give the shiny surface (note the default Roughness setting in this node is 0.2, set it to zero)
- A Refraction BSDF to give the effect of light refracting through the transparent material, and
- A Mix Shader to combine the two.

Ah, but how do we combine the two? Remember the **Fresnel factor** we talked about in the Blender Internal tutorial? Well, in Cycles, there’s a node for that!

In the Add SHIFT + A menu, bring up the Input submenu. In there, you will see a Fresnel node type. Add one of these. You will see it starts with a default IOR (Index of Refraction) value of 1.45, which is about right for glass.

Finally, connect together your little collection of nodes like this. See how the Fresnel factor controls the mixing proportion of the Glossy and Refraction shaders? A larger input factor to the Mix shader means a higher proportion of its second (lower) shader input, while a smaller factor gives a higher proportion of its first (upper) shader input. The higher output Fresnel factor corresponds to the regions closer to the edges of the object, where you want more reflection, while the lower value corresponds to the interior regions, where you want more transparency.

And finally finally, hit F12 to see what the result looks like.

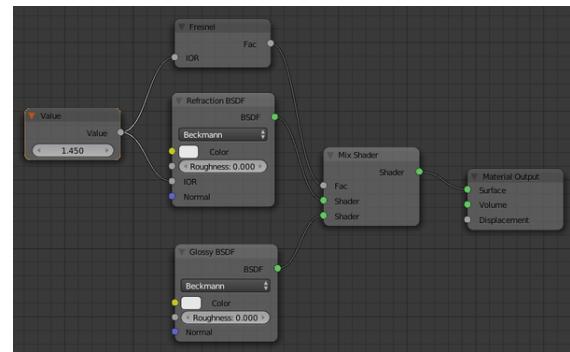
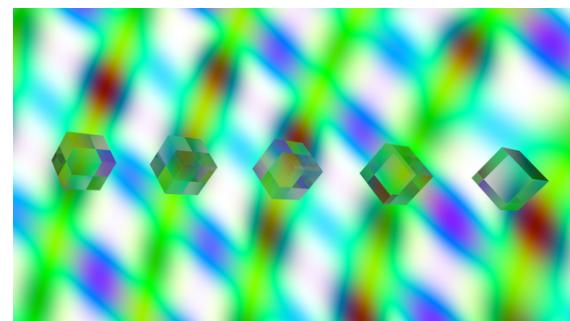
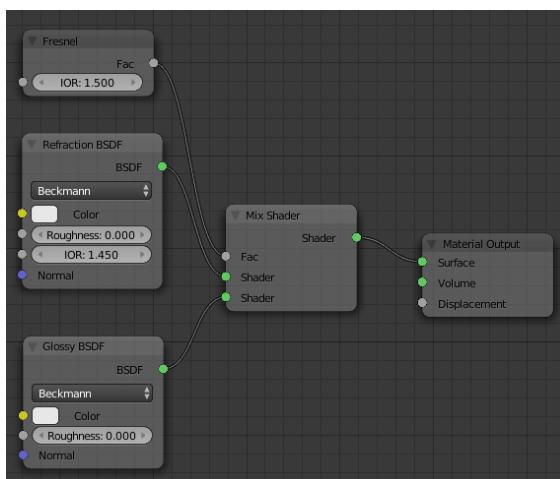
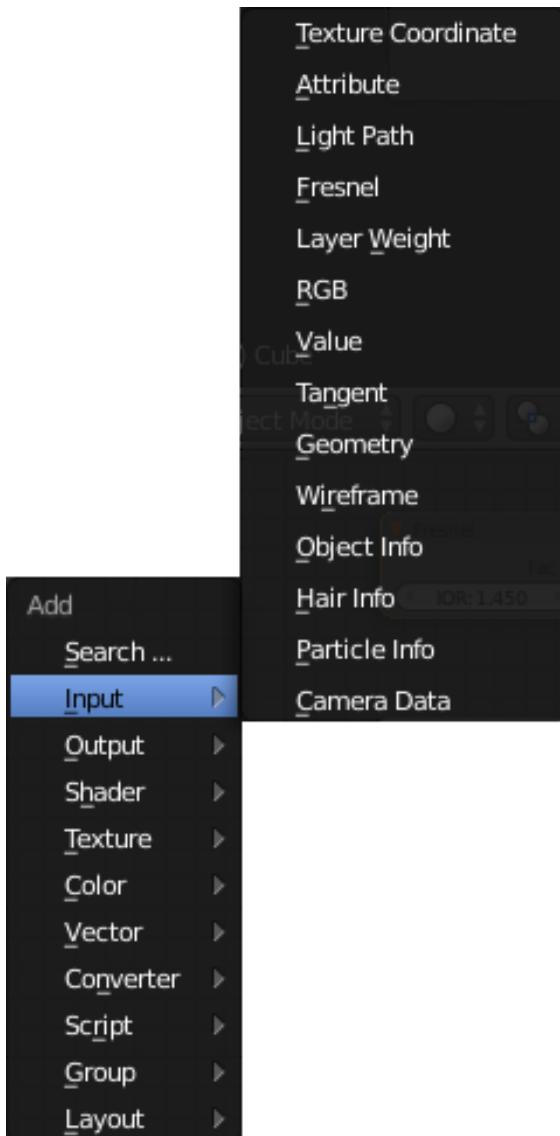


4.28.3 Adjustable Refraction

You may have noticed that the above collection of nodes includes *two* different fields labelled “IOR”: the one on the Fresnel node, and the one on the Refraction BSDF. The former governs the transition between transparency and reflectivity, while the latter governs the actual refraction through the material. In the real world, the two would always be the same, since they are the same physical quantity.

With the right IOR, this same node setup can be used to represent other transparent materials. Instead of having to set the value in two places, why not just set it in one?

To do this, add SHIFT + A another node, from the Input submenu, of type Value. This will feed a fixed value that you specify into any number of output terminals. Connect its Value output to the two IOR inputs, on the Fresnel and Refraction BSDF nodes. Here I have set its value to 1.45, which is a suitable value for glass.

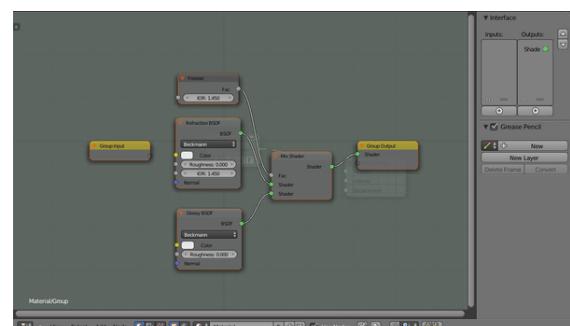


4.28.4 A Reusable Custom Shader

What if you wanted to have multiple transparent materials in a document? Would you need to go through all the above setup every single time?

Naturally you wouldn't. Instead, you can define a custom *node group*, and insert that into a Node Editor setup wherever you want, as many times as you want.

Start with the node setup as in the previous screen shot. Select RMB the Value node supplying the IOR value, and DEL etc it. Now look at the remaining four nodes, the Fresnel input and the three shader nodes (apart from the material output)? Select these, and only these, in any of the usual ways: press A once or twice to deselect everything, then RMB on the first node and SHIFT + RMB on the rest, or drag a Box selection around the desired nodes.

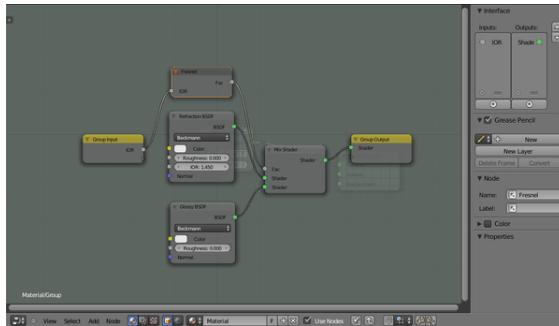


Now, with these nodes selected, select “Make Group” from the “Node” menu, or use the keyboard shortcut CTRL + G . Also press N to bring up the Properties shelf

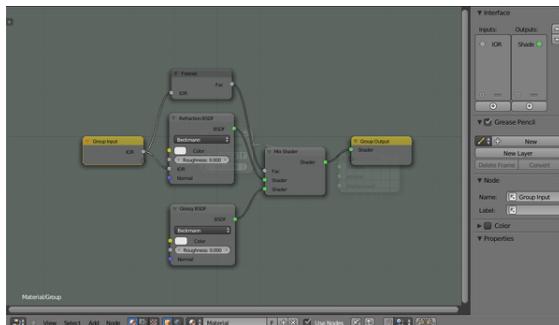
at the right side of the Node Editor window. The result should look like at right. Notice a couple of things:

- The greenish background indicates you are editing a node group, not the entire node setup
- The blocks with yellow headers, labelled “Group Input” and “Group Output”: these define the input and output terminals for the entire node group: inputs to the group appear as outputs on the Group Input block, and outputs from the group appear as inputs to the Group Output block. Any terminals within the group not connected to these will not be visible outside the group.

You can press TAB to switch out of editing the node group and back into editing the entire node setup, where you will see your group appear as a single block entitled “NodeGroup”. Try this if you want to have a look, then press TAB to get back into editing the group again, because we haven’t finished setting it up yet.

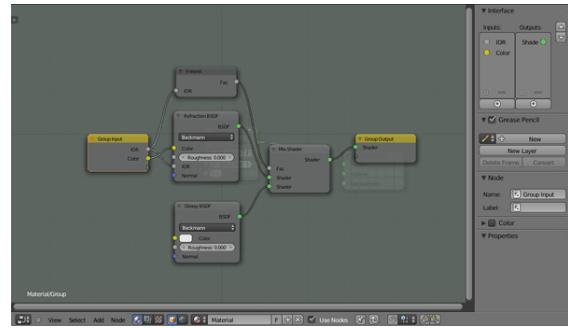


First of all, we need an input to specify the IOR value. Click on the IOR input terminal on the Fresnel node, and drag to the right edge of the Group Input node. Blender should automatically create an output terminal named “IOR” on the Group Input node, with a wire running to the Fresnel IOR input.

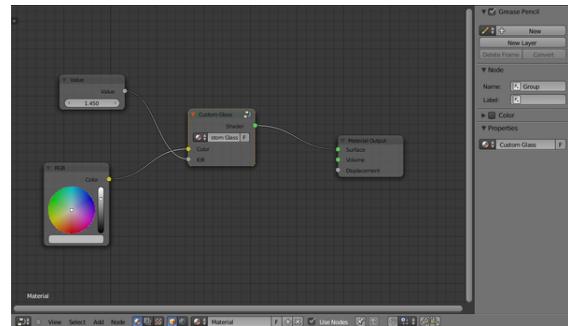


Now click on this Group Input IOR terminal, and drag another wire to the IOR input on the Refraction BSDF node.

Glass often has a colour. This colour affects light passing through the material, while glossy reflections remain uncoloured.



To create a colour input terminal for our node group, just drag from the “Color” input terminal on the Refraction BSDF node to the right edge of the Group Input node; as before, Blender will automatically add a new Group Input terminal and connection.

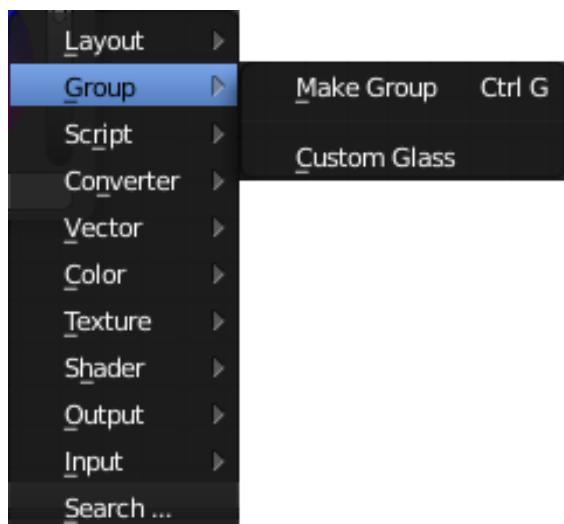


Now press TAB to finish editing your custom node group. This will take you back to the overall node setup, with your custom node still selected. Its name will be shown in the Properties panel in the Properties shelf at the right; change this from the default “NodeGroup” to something more meaningful, like “Custom Glass”.

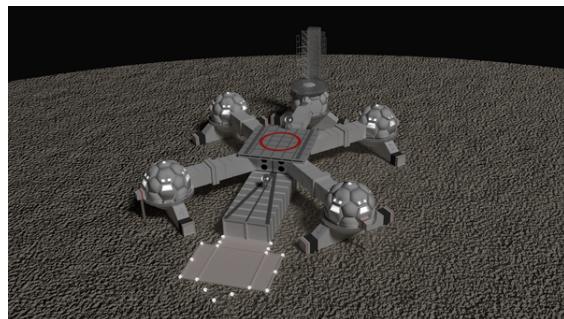
And now you can add back the input Value node you deleted, to supply a value to the IOR terminal. And similarly you can add an RGB node to feed into the glass colour. Or, since each value only needs to feed one input terminal, you can type them directly into the unconnected inputs on your Custom Glass node.

And furthermore, your custom node is now present in the Add Group menu, ready for you to insert somewhere else, in this or another node setup.

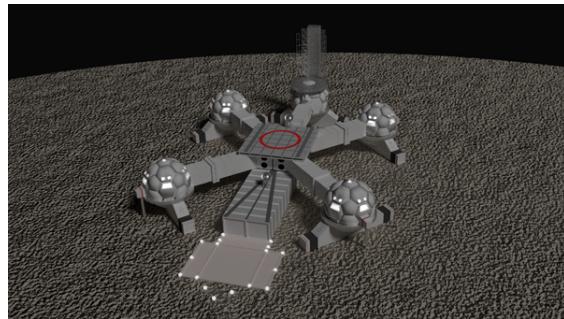
Note that the node definition itself is not duplicated when you insert another instance of your custom node. Each instance operates on its own separate input values and produces its own corresponding separate output values, but any changes you may subsequently make to the internal definition of the node group will automatically take effect everywhere you’ve already used it.



4.29 Dealing with Fireflies in Cycles



See the little white dots? (Click to view the full-size image, and look closely.)



Fireflies gone.

Cycles is a wonderful renderer, capable of some exquisite lighting effects. However, sooner or later, you will encounter the problem of **fireflies**, which are isolated bright pixels scattered over various parts of the image. Unlike simple *noise* (which can be reduced by increasing the number of render samples, or alternatively by simple filtering), fireflies are caused by numerical instabilities in the rendering computation, because after all computers can only calculate the formulas to a finite precision. They seem an inevitable consequence of trying to use realistic

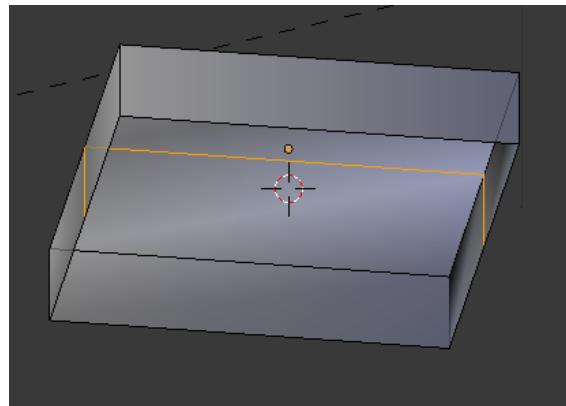
physics formulas to create our renders.

But are they so inevitable? In fact there are various tricks and tweaks we can use to keep them out of our renders. There are pros and cons to each technique (in terms of the amount of work involved and unwanted side-effects), so it is up to you to decide which to use and when.

Fireflies tend not to afflict the simplest renders. This means that it takes work to come up with a tutorial example sufficiently complex to exhibit them. If you already have one of your own renders which is being plagued with fireflies, and you came here hoping to find a fix, then great—you can skip ahead to the part with the fixes. Otherwise, please have some patience while we construct an example.

4.29.1 Making An Example

This example arose out of a real model I was trying to construct: a house with **louvred** windows made out of frosted glass. Some interaction involving light coming in through the panes of one window, bouncing around the diffuse walls of the room and out another window, together with the combination of shaders making up the glass material, led to the appearance of the fireflies. This example will cut the situation to the bare minimum I could come up with to exhibit the problem, namely a pair of adjoining rooms with two windows.

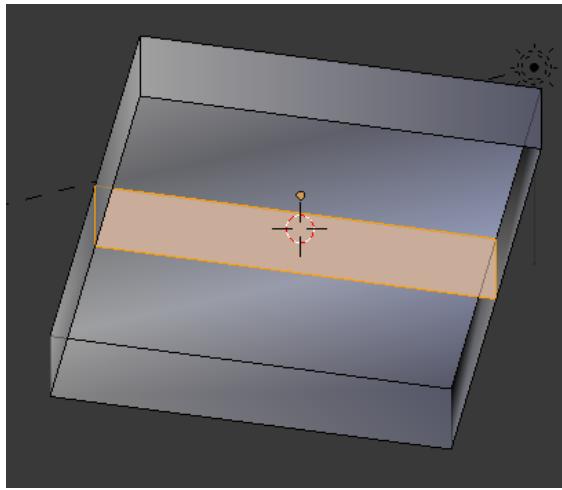


Loop cut to demarcate rooms (view from underside)

Start a new Blender document. Switch the renderer to Cycles. Select the default cube, and TAB into Edit mode. Scale the cube out horizontally a bit so the room is not so narrow and tall. Switch to face-select mode. Select the bottom face and DELete it. I found that the room needed a ceiling to show the fireflies, but not a floor, so why not get rid of it?

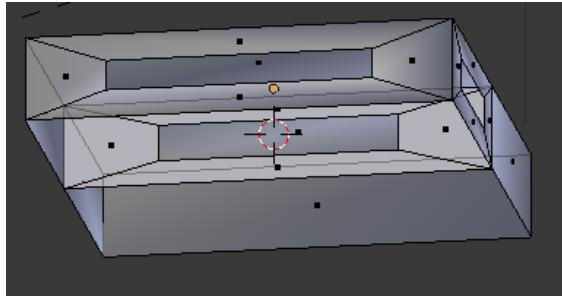
Give your cube a Solidify modifier, so the walls have a nonzero thickness. The exact thickness doesn't matter.

Next, make a single loop cut CTRL + R around the sides of the shape to define the boundary between the front and rear rooms. Then press F, and the vertices of the new



Wall filled in between rooms.

loop should be joined to form a new face which is the wall between the rooms.

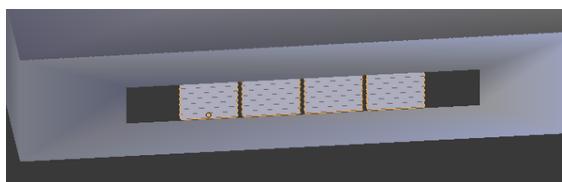


Windows and doorway cut into walls.

Now we need to make holes for the windows, and for a doorway between the rooms. Still in face-select mode, select the front face (the one you see in NUM1 view), and I inset it a little way. If you inset it too far, you may find the height gets too small, so after the initial insetting, select just the middle face and scale it horizontally S + X to make a reasonable-looking window.

Do the same with the wall on the right side of the front room, and also on the wall between the rooms. A *real* doorway would extend to the floor, but our simple hole in the wall is sufficient for this exercise.

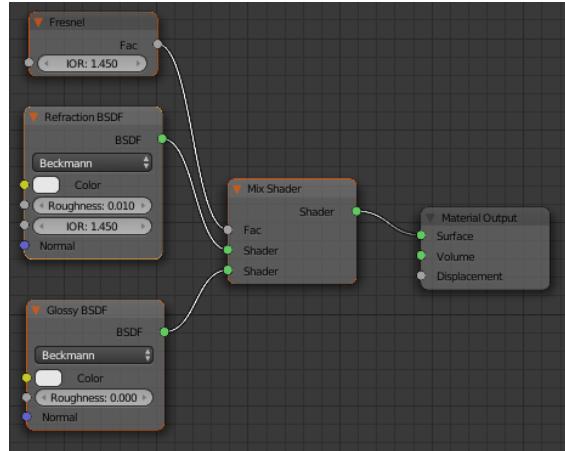
Add The Windows



Front window with louvres.

Add a new horizontal plane. Shrink it along the Y-axis

to 20% of its length along the X-axis. Add a couple of array modifiers: one along the Z-axis with a constant offset of something like 0.2 and a count of about a half a dozen, and another along the X-axis with a constant offset around 2.2 and a count of 4. Position your resulting array of louvres within the hole for the front window.



Recipe for fireflies

Before going any further, we mustn't forget to make the windows out of frosted glass. Assign a new material to your window object. Bring up the Node Editor, and set up a material for the window glass as at right. Set the roughness on the Glossy BSDF to zero, but note the small, but nonzero Roughness setting on the Refraction BSDF: I deliberately tuned this to maximize the production of fireflies. A zero setting will make them go away, but that defeats the point of this exercise, doesn't it?

(The room walls can be left at the default diffuse material assigned by Blender.)

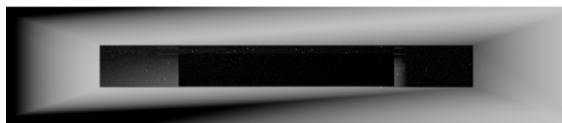
After setting up your window material, duplicate your louvres object and rotate and position the copy within the hole for the side window. Leave the hole between the rooms empty (it *is* supposed to be a doorway, after all).

Completing The Recipe

As the final part of our recipe for fireflies, change the type of the Lamp object to Sun, and change its angle to close to horizontal so it will shine in through the side window. Give it a strength of 3. Position the camera to look directly into the front window. Also increase the number of render samples to 100; that should be sufficient to keep down the noise.

So, in summary, we have sunlight coming in low through louvres in a side window in the front room, bouncing around between front and back rooms, and exiting through louvres in the front window, the louvres all being made of lightly-frosted glass.

If all goes well, hitting F12 to render should produce an



Fireflies! (Click to examine more closely.)

image like at right.

4.29.2 Fixing The Problem 1: Renderer Tweaks



Relevant renderer settings

OK, now that we can create the problem, how do we solve it?

Probably the first things to try are one or two renderer settings. Look for the Sampling and Light Paths panels

in the Render context, as at right. In the Sampling panel, look for the “Clamp” field. If nonzero, this imposes a maximum value on the light-intensity calculations at each pixel. Normally 1.0 corresponds to full white, but some points can be brighter than this. Try changing the default 0 value to, say, 3, and rerender.

In our simple example, this should be enough to fix the fireflies. However, I have come across a case or two where this reduced the intensity of some bright indirect lighting, leading to a more subdued appearance for the overall render.

So let's try the next thing. Set the Clamp back to 0, and

this time look in the Light Paths panel, for the “No Caustics” checkbox. Caustics can be an important part of realistic lighting effects, but if your scene doesn't need them, disabling them can help get rid of fireflies, and perhaps speed up the render as well. However, the tooltip for this checkbox warns that the result can be a darker image.

In our simple case, checking this box *does* work.

The third renderer setting to try is the field just below the No Caustics checkbox, labelled “Filter Glossy”. This applies deliberate blurring to certain kinds of light bounces, sacrificing a bit of lighting accuracy for the sake of avoiding fireflies. And again, in our case, this option works: setting it to a value of, say, 1, does indeed fix the fireflies.

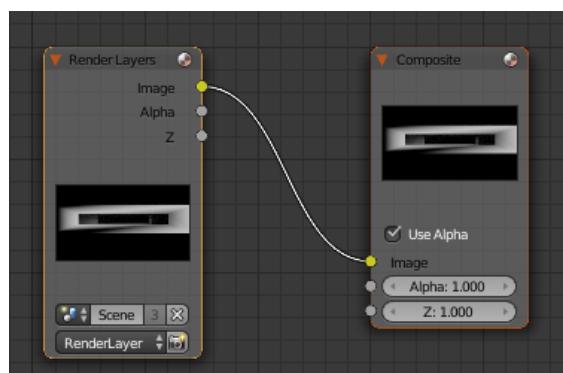
But, at some point, you may find that none of these three techniques will work for you: they may make no difference to the render, or may cause unacceptable loss of quality in places. After all, being settings that apply to the render as a whole, they can be something of a blunt instrument.

So for our next fix, we will need to resort to the compositor.

4.29.3 Fixing The Problem 2: The Despeckle Filter

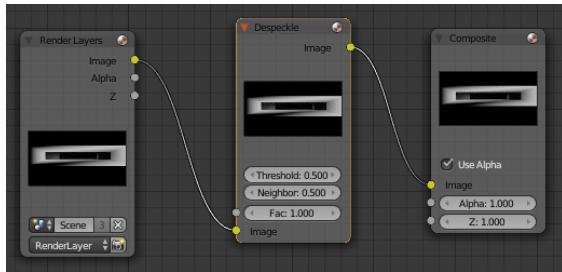
The approach to fixing the fireflies problem is to apply some kind of processing to the rendered image. If you are familiar with image-editing programs like Gimp or Photoshop, you will have come across the *despeckle filter*, whose job is precisely to smooth out random dots that don't fit in with the rest of the image. In principle you could use an image-editing program to touch up your renders (and some people producing professional-quality renders do), but what happens if you are rendering an animation? Imagine having to fix up hundreds or thousands of frames by hand!

Blender, too, has a bunch of image-processing operations available in its compositor, including a Despeckle filter. And the nice thing is, these operations can be applied automatically after rendering each frame.



Open up a Node Editor window in your Blender doc-

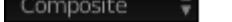
ument (you can reuse the Timeline  window at the bottom of the default window layout, just make it a bit taller). Look for the icons  indicating what type of nodes to edit: click on the one showing little layers or sheets, representing the compositor (in the middle in Blender 2.68, on the left in earlier Blender versions). Immediately you should see a “Use Nodes” checkbox appear to the right of these icons: check that to enable node-based compositing. You should get a default node setup like at right.



Move the two existing nodes a bit further apart, to make way for a new one in-between. Press SHIFT + A to bring up the Add menu, and look in the “Filter” submenu for the “Despeckle” entry. Move the new Despeckle node into the gap you previously cleared, and if you position it right, it should automatically connect itself into the chain like this.

To see the effect of your changes, switch the 3D view window to an Image Editor window. Make sure the image being shown is the “Render Result”. Next to the menu for selecting the image slot (which should be showing “Slot 1”), there is a menu for selecting which layer of the image



to show: . “RenderLayer” is the raw output from the renderer, while “Composite” is the result after going through the compositor. By switching between these two, you should see that the default Despeckle filter settings have removed some of the fireflies, but not all. I found if I reduced the “Threshold” filter parameter from its default 0.5 to something like 0.1, then all the fireflies did indeed disappear.

What The Despeckle Filter Does

The despeckle filter is designed specifically to get rid of isolated anomalous pixels. This is unlike blur filters, which average a bunch of pixels together, and can spread the firefly across a larger area instead of removing it.

But what do those “Threshold” and “Neighbor” parameters actually mean? The following explanation was put

together from studying the Blender source code, mainly [source/blender/compositor/operations/COM_DespeckleOperation.cpp](#).

The target pixel being processed is compared with its eight neighbours. However, these neighbours are not treated equally: each one is given a *weight* that governs its relative importance to the computation. The horizontal and vertical neighbours each have a weight of 1, while the diagonal neighbours are each given a lower weight of $\sqrt{\frac{1}{2}}$ (approximately 0.7). This reflects the fact that they are slightly further away along the diagonals, therefore they have correspondingly less influence on the outcome.

For each neighbour pixel, the “Threshold” input parameter is compared against the difference between each of the neighbour’s R, G and B components (normalized to 0 being black and 1 being maximum intensity) and those of the target pixel; if any of the corresponding pairs of components have a difference exceeding the threshold, then the neighbour pixel value is accumulated onto an average (weighted by neighbour weights) that the code calls `color_mid_ok`—the presumed “correct” value of the target pixel, if you like. All neighbour pixels, whether within or outside the threshold, are accumulated onto another (also weighted) average called `color_mid`. The value *w* is the total weight of all neighbours that contributed to `color_mid_ok`.

Then the value of *w* is divided by the total weights of all neighbouring pixels ($4 \times 1 + 4 \times \sqrt{\frac{1}{2}}$), to produce a fraction in the range [0.0 .. 1.0], and the resulting fraction compared to the “Neighbor” input parameter. Also the difference between the components of `color_mid` (the average of all the neighbours) and those of the original target pixel are compared to the “Threshold”. If either comparison returns less than or equal, then the pixel is considered not to need correction. But if *both* comparisons are greater, then the target pixel is adjusted from its initial value towards the `color_mid_ok` average, according to the “Fac” input socket value. If “Fac” is 1.0 (the default), then the pixel is completely replaced with `color_mid_ok`; otherwise it becomes the corresponding blend of this and its original value.

“Fac” can be used to fine-tune the effect of the Despeckle. For example, if this was fed from an inverted edge-detection (“Kirsch”) filter, then it can be used to reduce the effect along sharp edges, which are otherwise likely to confuse the filter.

4.30 Fireflies in Cycles, Continued

You previously read about techniques for controlling fireflies in renders done with Blender Cycles, and how, when all else fails and tweaking renderer parameters doesn’t help, it is time to roll out the Despeckle filter. Sometimes a despeckle applied across the entire image may be too blunt, leading to unwanted effects in parts of the image

where there are no fireflies.

But never fear. There are yet more options in Blender's compositor that we have yet to explore, to give us even finer control over the application of despeckling, and hopefully avoid the final, desperate measure of hand-retouching (shock! horror!) individual blemishes in an image-editing program.

The basic principle is, fireflies are caused by specific interactions of types of materials and lighting, hence they will only occur in particular parts of the image. By separating out the problematic materials/lighting cases, we can selectively apply the filter just to them, minimizing side-effects on parts of the image that look just fine.

We will present a way to do the separation.

4.30.1 Finer Despeckling: Separate Lighting Passes

Separating The Lighting Passes

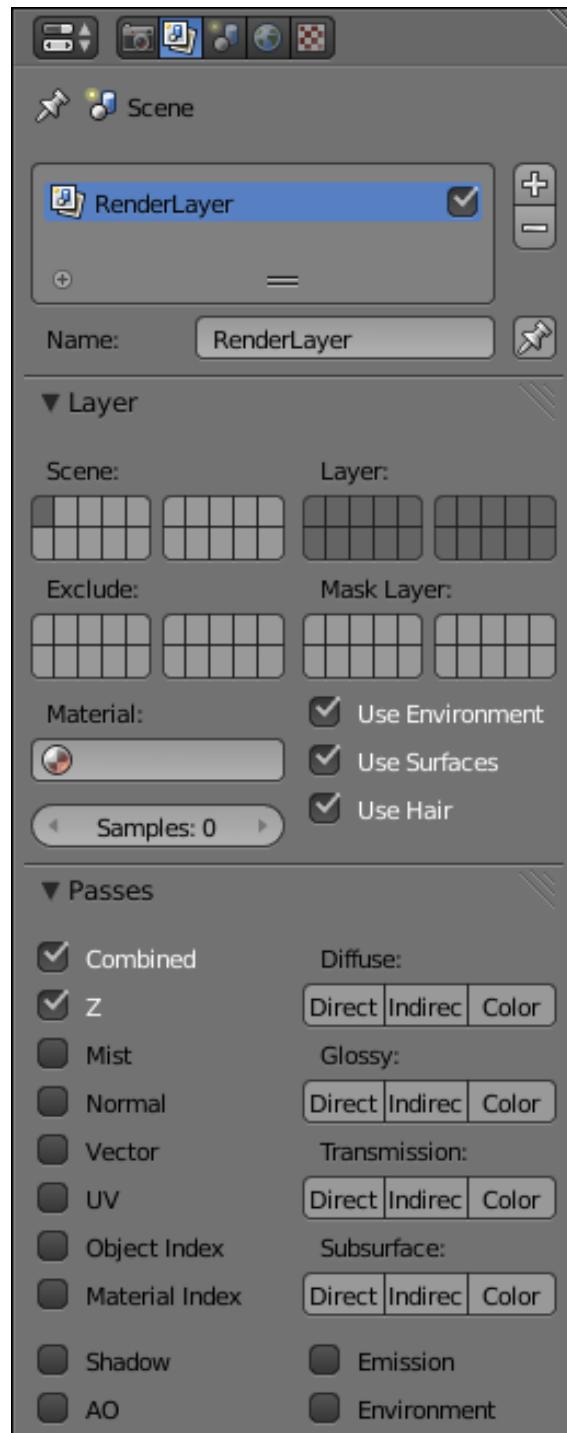
The Cycles renderer has the capability to separate out different kinds of light paths into separate *passes* for individual processing. Consider the diagram on the Blender wiki, showing how the direct lighting, indirect lighting and colour calculations are broken down by diffuse, glossy and transmission light paths, before being combined together with emission and environment lighting. We can split out all these passes, and use compositor nodes to recreate the standard combination as described there. But we can also intersperse our own processing onto particular passes, leaving the others untouched.

To tell Blender that we want to extract the light passes, go to the Render Layers  context in the Properties  window, and look for the Passes panel. Notice the four groups of 3 buttons each for "Direct", "Indirect" and "Color" grouped under "Diffuse", "Glossy", "Transmission" and "Subsurface"? (Depending on your Blender version, the "Subsurface" group might not be present.) Click all these buttons, and also check the "Emission" and "Environment" boxes just below.

Now if you go back to the compositor window, you should see that the RenderLayers node has sprouted a whole lot of extra output terminals, one for each of the buttons/checkboxes you clicked.

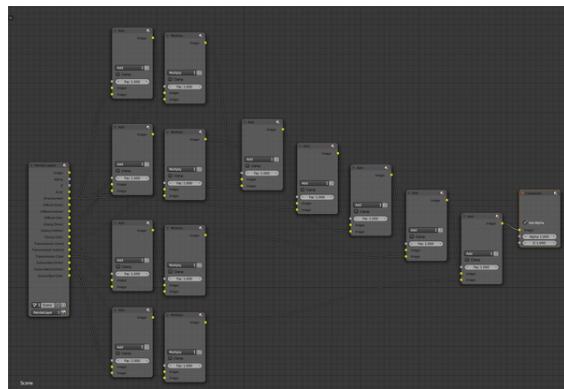
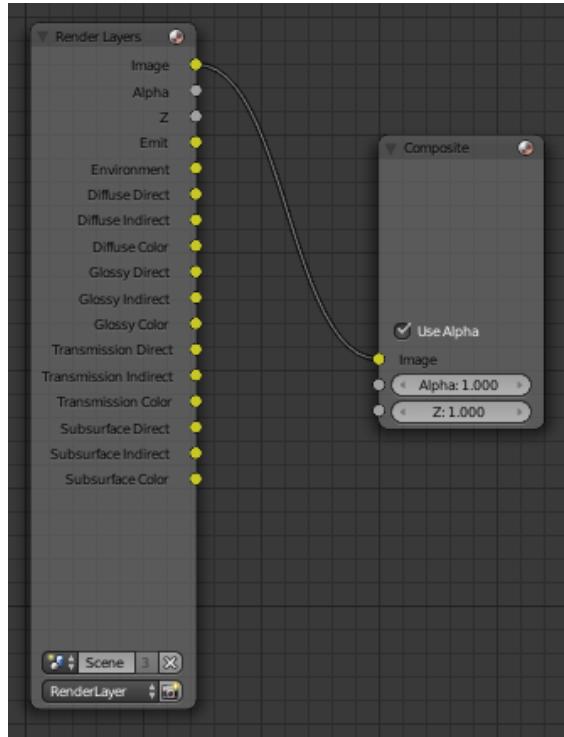
Next, we want to recombine these to mimic the behaviour that Blender provides by default. If you go back to that Blender Wiki diagram above, you will see that, for each shader type (diffuse, glossy etc), the direct and indirect light passes are added together, and the sum is then multiplied by the colour calculation. The result from all the passes is then added together.

The Blender compositor has standard node types for adding and multiplying colours. Armed with this infor-



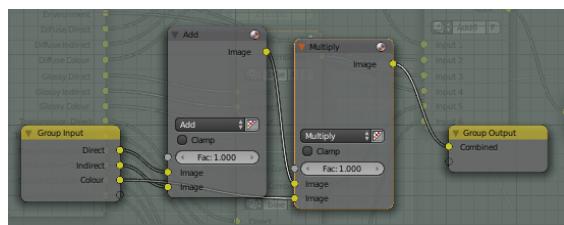
mation, you might immediately set to work putting together an intricate node setup to implement the formula depicted in that diagram, perhaps ending up with something like at right.

Show this to a seasoned computer programmer, and they will perform the gesture known as a "facepalm". Yes, technically this will work, but it is quite complicated-looking and error-prone. Supposing you get a connection wrong: how would you find it? How would you debug your node program?



How not to do it: notice the repeating patterns?

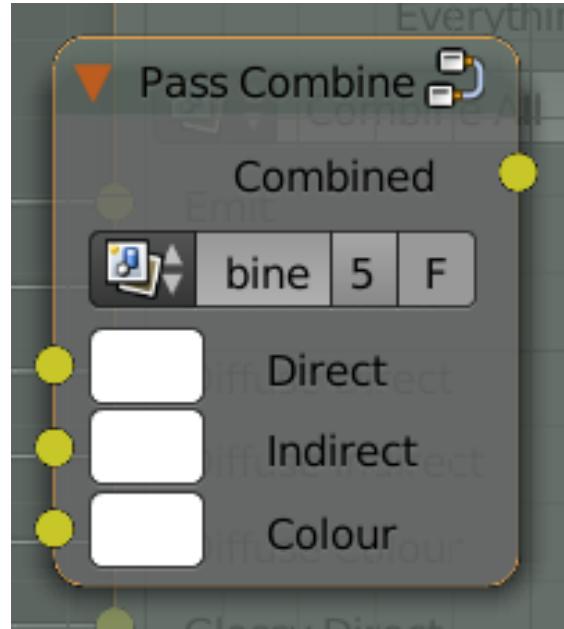
For, make no mistake, **nodes are a programming language**. Putting together a node setup is very much like writing a computer program. One of the important principles is **don't repeat yourself**. That is, if you find yourself doing the same sequence of steps more than once, it makes sense to spell it out once, and then next time just tell the computer to do the same thing again.



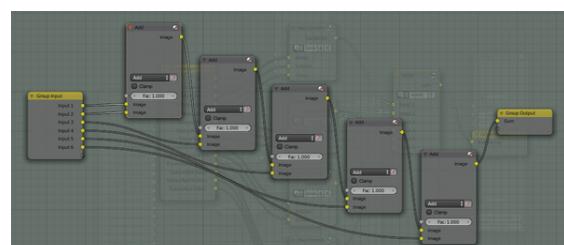
So let us see how a programmer would approach this same

problem. To start with, consider the most basic operation, which is: add the direct and indirect components, and multiply by the colour. So create a pair of Mix nodes, one to do the Add and the other to do the Multiply, and wire them appropriately. Turn them into a group **CTRL + G**, and add group inputs and outputs as in the diagram.

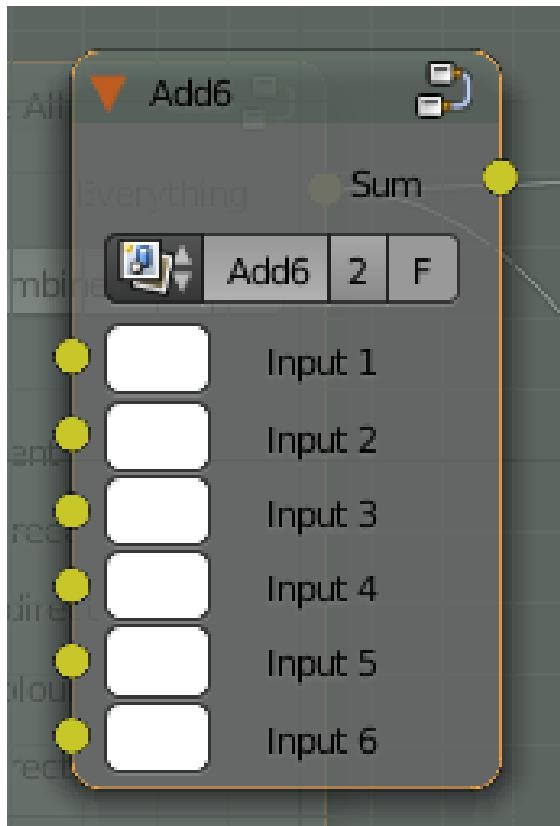
To change the names of the inputs and outputs, you can select them in the list that appears in the Properties shelf N in the Node Editor window, then type a new name in the appropriate field just below the list.



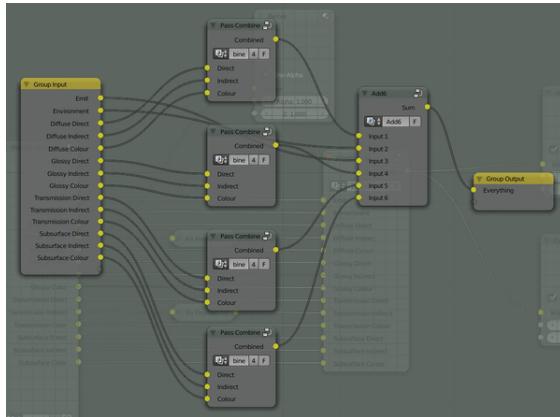
Having done all the wiring, TAB out of the group, so you can give it a name. Here I call it “Pass Combine”. See how naming the inputs and outputs helps make it clearer what the node is for? And if you want to see the details, just TAB into the node for more. This technique is called *abstraction*, where you start by seeing only the overview, but you can progressively peel back the layers to see more and more detail, but only where you want to. This is an important technique for dealing with complexity, which is what programmers do every day.



We will need four instances of this group, plus another bunch of Mix nodes to add them all together. We are adding 6 quantities, so we will need 5 Mix nodes. Again, I make a group to combine all these adders, just to make things neater.



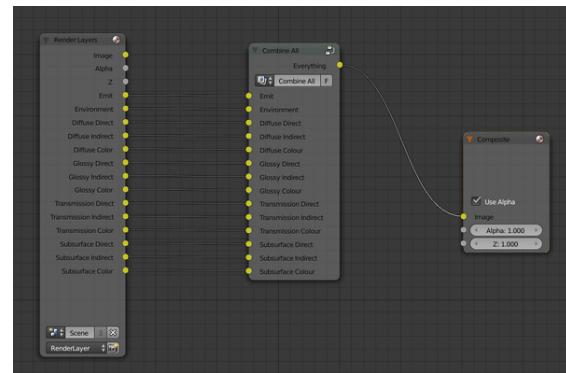
I call the resulting group “Add6”. Blender only provides nodes to add two quantities at a time, yet here we have made our own custom node that can take 6 inputs at once. Neat, huh?



So now we combine our building blocks into a new building block that computes the combination of all the passes, and looks like this.

Note the ordering of the group inputs, to ensure a nice and neat arrangement when we connect this node up below. You can rearrange the inputs by selecting one from the list in the Properties Shelf N and using the up- and down-arrow buttons next to the list to change its position.

And then when we connect that building block (here

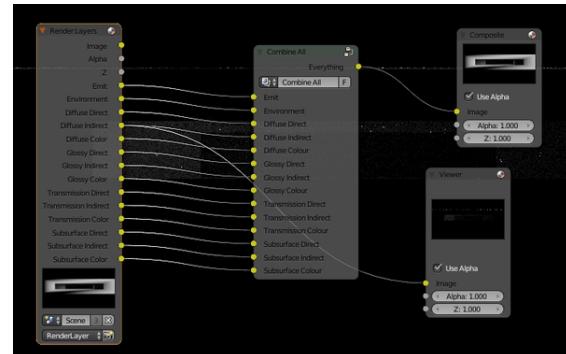


called “Combine All”) into all the RenderLayers outputs, our complete node setup now looks like this.

Compare this with the original simple-minded recreation of the pass-combining algorithm: which would you rather be presented with when you open a Blender document?

Fixing The Fireflies

Anyway, now that we have separated the lighting passes, such that we can recombine them to produce the original image anyway, how do we actually go about fixing the fireflies?

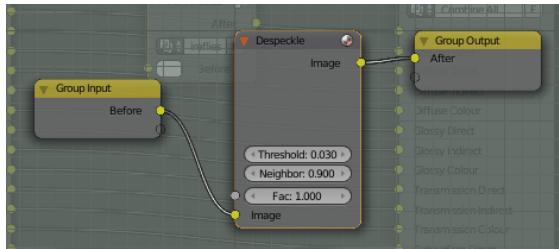


Diffuse-indirect pass produces fireflies

First we have to isolate which passes are producing the fireflies. Click the “Backdrop” checkbox in the header of the compositor Node Editor window. Now, CTRL + SHIFT + LMB on the RenderLayers node, and you should see a new Viewer node automatically appear and connect itself to the first output terminal on that node. At the same time, the background of the window changes to show the image produced by that output terminal. The first terminal should be showing the complete image, including the fireflies.

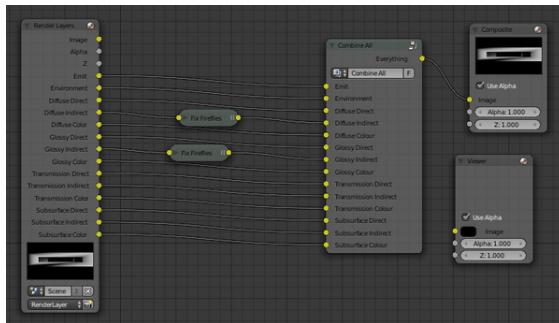
Keep pressing CTRL + SHIFT + LMB , and the Viewer node will connect to each output terminal in turn. Some may be all black or all white, but don't worry, just keep going. When you come to the Diffuse Indirect output, you should see something like at right, mostly dark, but with noticeable fireflies. Make a note and keep going; I also

found that the Glossy Indirect output was another source of problems in this case.

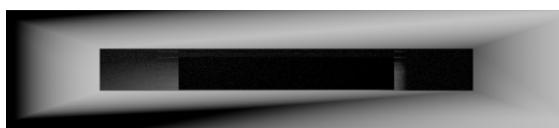


So we need to insert a Despeckle node between the Diffuse Indirect output from the RenderLayer node, and the corresponding Diffuse Indirect input on our Combine All node. But instead of just making a simple Despeckle node, let's make a custom node group. Why? Because we'll need it in two places. So rather than having to set it up twice, just do the setup once, and insert two instances. That way, we can adjust the despeckle settings in just one place, and have it automatically take effect wherever there is an instance. Also, maybe we want to try other filters in combination with or instead of Despeckle; again, we would only need to make the change in one place, and it would automatically affect all instances.

“Don’t repeat yourself”, remember?



So here we are, with a couple of instances of our custom “Fix Fireflies” node group inserted in the most troublesome light paths.



Fireflies gone!

And here is the resulting render.

4.30.2 Conclusion

Fireflies can be troublesome, particularly because there is no 100%-foolproof way of getting rid of them. After all, if there were, why not just build it in as an automatic

part of every render? This pair of pages has presented a range of techniques for dealing with them, from simple to complex. Of course it makes sense to try the simple ones first, and only if they’re not good enough, is it worth expending effort on something more complex.

4.30.3 See Also

- This [video](#) by Bartek Skorupa, which was the inspiration for this page.

4.31 Procedural Eyeball in Cycles

You [previously learned](#) how to model an eyeball using just procedural textures with the Blender Internal renderer. This page will redo the exercise using the Cycles renderer.

Note: The BI version of the tutorial relied on a quirk of the Musgrave texture, in that a certain combination of settings produced a nice pattern of radial ridges for modelling the iris of the eye. While Cycles also has a Musgrave texture, I was not able to reproduce the same effect. However, Cycles has a much more powerful material node system, and in particular by transforming to polar coordinates and applying a radial scaling, I was able to get a suitable pattern of radial ridges using just a simple Noise texture.

Why Procedural?

Rather than mess around with intricate manipulations of procedural texture generators, the previous tutorial [Creating Pixar-Looking Eyes](#) used an image for the iris texture. This simplifies some things, but it has its own drawbacks:

- How do you create this image texture? Do you hand-paint it? If the resolution is not high enough, or the patterns or colours are not right, you must redo it.
- It’s a separate file you have to remember to include with your project. The Cycles renderer does not (yet) support packing images into the .blend file.

As you will see below, a procedural node setup gives you plenty of opportunities to tweak parameters, to produce effects that are subtly (or not-so-subtly) different. This allows for a great deal of variety, with very little effort—once you have worked out the node setup, of course.

4.31.1 Creating the Mesh

The modelling part of the exercise is exactly the same as in the BI version. If you already have a mesh that you did

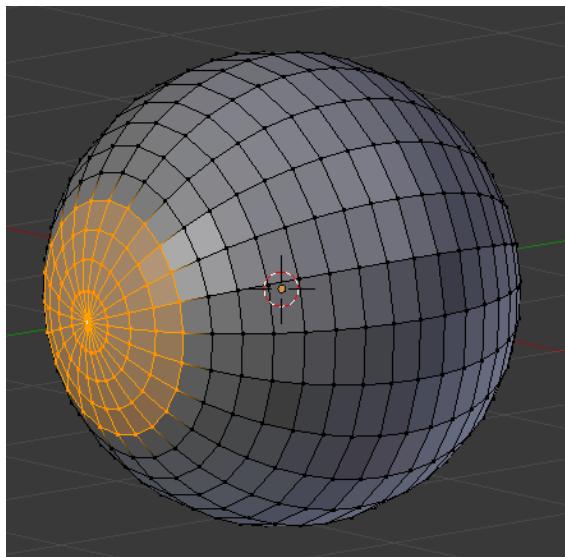
for the BI tutorial, feel free to reuse that and skip to the next section. Otherwise, continue reading.

The mesh will consist of two modified UV spheres, one just slightly inside the other. The outer mesh will have a material that is both transparent and reflective; at the front of the eye, this will represent the *cornea* (where the light enters the eye), while over the rest of the eye, it will add shininess to the eyeball. The inner mesh will make up the remaining three major parts of the eye that are visible from outside:

- the *sclera*, the white part of the eyeball
- the *iris*, the variously-coloured ring of muscle that surrounds the hole that actually lets in the light, and
- the *pupil*, the hole where the light goes into the interior of the eye.

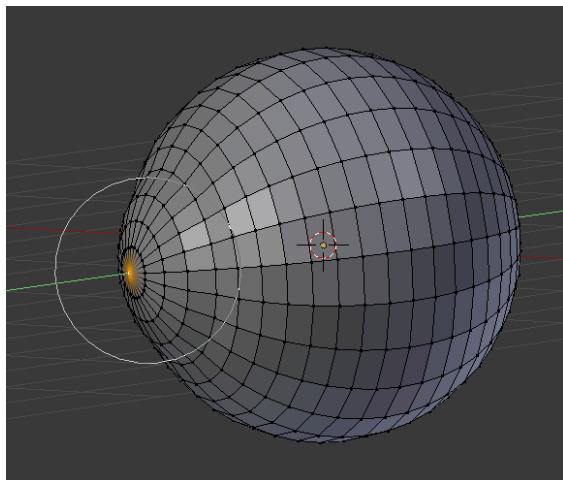
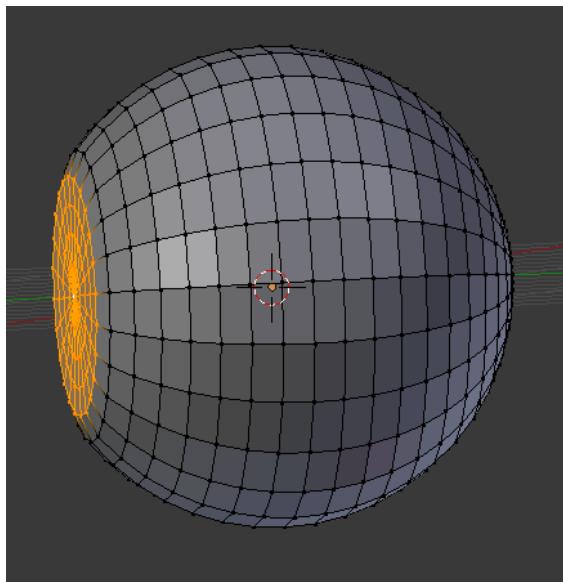
Start in Object mode. Get rid of the default cube. SHIFT + A dd a new UV sphere mesh; 24 rings and segments should be sufficient. Switch to front view NUM1 , and check the “Align to View” box that should have appeared in the panel at the bottom of the Tool Shelf. This will be the inner mesh.

TAB into Edit mode; make sure the entire mesh is selected; SHIFT + D uplicate it, and S cale it slightly up to create the outer mesh. (You may find it easier to work in wireframe Z mode.) H ide the outer mesh you just created for now, to make it easier to work on the inner one.



Select RMB the single vertex at the front of the inner mesh. Now press NUM+ four times to extend the selection to include the first four rings of vertices out from this front vertex. The selection should look like at right.

Now flatten the selected vertices with the sequence S Y 0KEY ENTER . Next, reduce the protrusion of the resulting flattened disc with G Y and moving it a little closer to the rest of the eyeball. The result should look like at right.



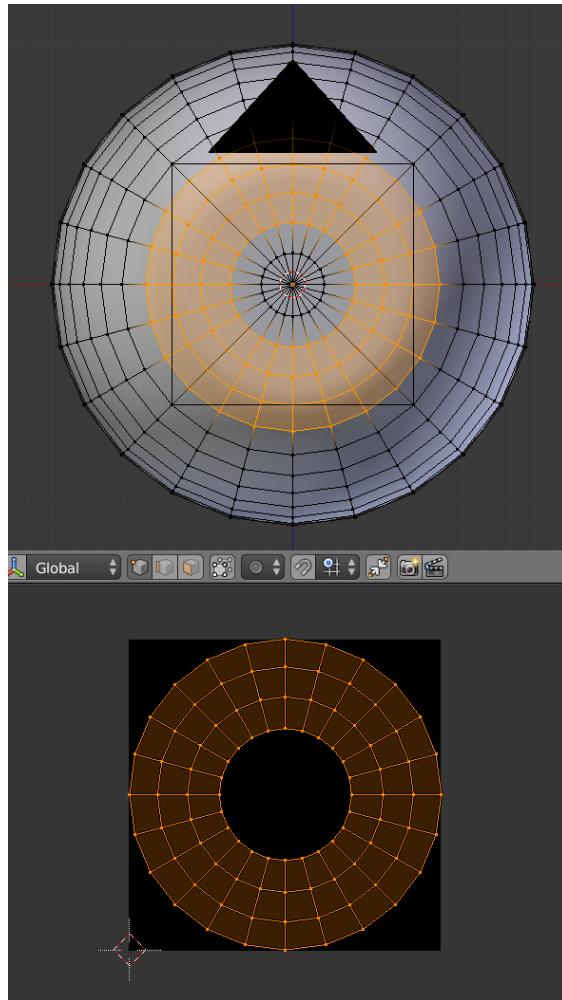
This will do for shaping the inner mesh. Now to do the outer one. Unhide it ALT + H . Select RMB just the frontmost vertex. In the Mesh menu, set Proportional Editing to “Connected” and Proportional Editing Falloff to “Root”. Now move the selected vertex forward with G Y , and use the mouse wheel at the same time to adjust the proportional editing influence until you make a bulge that is about the right proportions for a human cornea.

OK, that’s the modelling done, now on to the fun part—creating the materials in Cycles.

4.31.2 UV-Mapping the Iris

The iris pattern needs to be correctly located so that the radial striations are symmetrically arranged around the centre. To control the placement of the iris texture, we need to set up a UV map.

Set your layout to have both a 3D view and a UV/Image Editor view, as at right. Create a new image in the Image



Editor; this will be just a dummy for the UV unwrapping, so its size and contents do not matter.

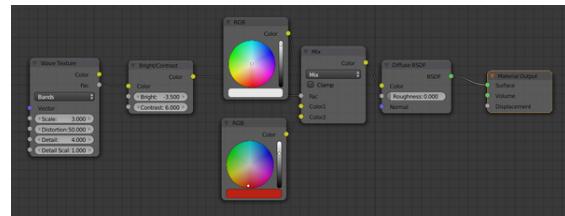
Make sure you are in Edit mode in the 3D view. Hide the outer mesh, to reduce the clutter. Switch to frontal view NUM1. Make sure (A once or twice) nothing is initially selected. Select the rings of vertices that will make up the iris, as at right. Bring up the U nwrap menu, and select “Project from View (Bounds)”. This should unwrap the selected vertices, exactly as they look in the frontal view, and place them neatly within the bounds of the dummy image, as in the lower window in the screenshot at right.

4.31.3 Creating the Materials

The Cornea

For the outer mesh, you can set up a material similar to the one previously discussed in [A Glass Material in Cycles](#). Only, instead of an IOR of 1.45 (glass), use a value like 1.33 (water), since water is the main constituent of the transparent tissue (as indeed of all human tissue).

The Sclera

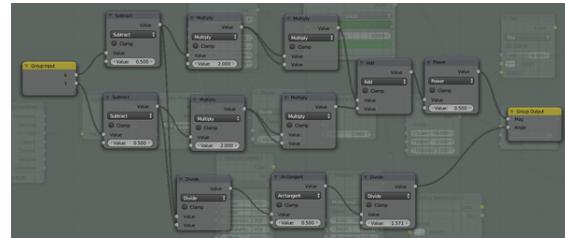


The material for the white of the eye needs a little tracing of red blood vessels, but not too much. Here the network of blood vessels is provided by a Wave texture node with a very high distortion setting. This is then put through a Bright/Contrast node: enhanced contrast to sharpen the edges, and reduced brightness to make the blood vessels very pale. This value is then fed to a Mix RGB node to mix between the white of the main part of the eyeball and the red of the blood vessels.

The original tutorial added some shininess to this material. However, since the cornea material is already shiny, and already overlaying the entire eyeball, it seemed redundant to have shininess here as well.

The Iris

This is the fun one, because it will require the most intricate Cycles node programming.



As previously mentioned, we have to convert some texture coordinates from **rectangular** to polar coordinates. The node group at right will achieve this. It looks quite complicated! However, it becomes easier to understand if you take it step by step:

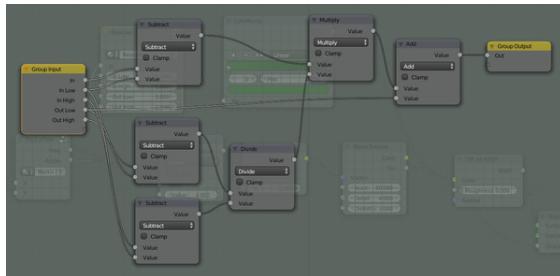
- The two rows of 3 nodes each at the upper left (each containing a Subtract followed by two Multiply nodes) take the incoming X (upper row) and Y (lower row) values, subtract 0.5 to convert the range from 0..1 to $-0.5..+0.5$, multiply by 2 so the range is now $-1..+1$, and takes the square of each value. These are then added and the square root (Power of 0.5) of the result taken, to give the distance of the point from the origin, in the range 0..1. In other words, it is computing the equation $r = \sqrt{x^2 + y^2}$. This is the Mag output from the node group.

- The bottom-most row of nodes divides X by Y (actually it should be Y by X, but that doesn't matter for our purposes) to compute a slope, and then uses the arctangent function to get the angle of this slope. This value is in radians, so it is divided by 1.571 ($\frac{\pi}{2}$ or close enough) to convert it to the range $0..1$. This becomes the Angle output from the node group.



Now we use the above node group in *this* node setup. Note the “Separate RGB” and “Combine RGB” nodes: they are not separating/combining RGB colour components, but *XYZ coordinate components*. In both cases, the components are scalar reals, so mathematically speaking, this (ab)use of the separate/combine RGB functions is perfectly all right. Note also how we ignore the B component (i.e. the Z coordinate), since we are only dealing with a two-dimensional texture.

After separating out the X and Y values, these are fed through the Rectangular→Polar conversion node group we created above. But then note how the Mag value is put through a division by 4 to stretch out the texture radially: this is the secret to getting the radial ridges in the iris texture. After the coordinate components are recombined, they are used to control a Noise texture, which then gives a bump map to the iris material.



But we’re not done yet. There is still some more node magic to wield, to get a radial gradation in colour from the centre to the edge of the iris. To build that, we will need another node group, as at right. This one is called “Rescale”: it takes a value in an input range (defined by the “In Low” and “In High” inputs) and linearly transforms it to an output range (defined by the “Out Low” and “Out High” inputs). Or, using mathematical symbols x for the input value, y for the output value, and x_0, x_1, y_0 and y_1 for In Low, In High, Out Low and Out High respectively, this node group implements the transformation $y = (x - x_0) \frac{(y_1 - y_0)}{(x_1 - x_0)} + y_0$.

And now, here’s how we add a use of the Rescale node group to our iris material node setup. We use it to transform the Mag polar coordinate (distance from centre),

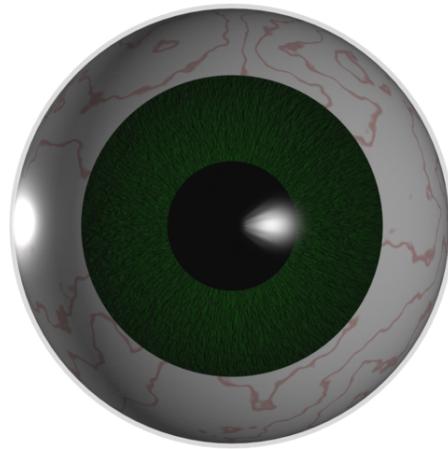


and then use that to drive a colour ramp that actually controls the diffuse colour of the iris. Here I have done a gradation from light green to dark green; feel free to substitute whatever eye colours and gradations you choose.

The Pupil

Here I just used a simple black diffuse material. The original tutorial added some subtle suggestion of red blood vessels, similar to the sclera; you could do that here if you want, but I thought the effect was too subtle to see, so I didn’t bother.

4.31.4 The Final Result



And here is what the final render might look like.

Parameters to Tweak

The above node setups are quite complex, but they offer many opportunities for experimentation.

For example, in the Sclera material, try tweaking the Bright/Contrast settings, to see how that affects the result. Make the red blood vessels more prominent, and the eye looks bloodshot!

In the Iris material, you can change the eye colour (of course). But also try changing the radial striations: see

how they look if they are larger. Why did I put the Rescale node there? It was to let you increase or decrease the effect of the colour ramp, without having to edit the ramp itself.

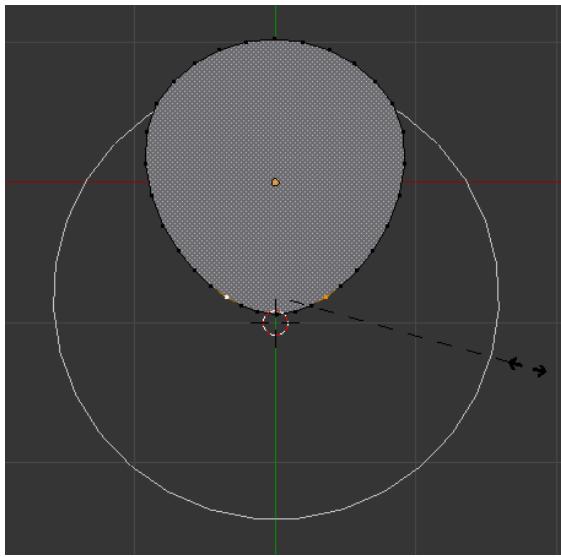
4.32 Introduction to Freestyle

New in Blender 2.67, the **Freestyle** renderer adds a whole range of *non-photorealistic* effects to the Blender Internal renderer. It seems in some ways like the decades-long quest for photorealism in computer graphics has come full circle, and people are now deliberately seeking how to achieve more cartoony or line-drawing-style effects!

4.32.1 A Simple Freestyle Example

Start with a new Blender document. Delete both the default cube  and the default lamp. In the World  context, check the Environment Lighting box and set its energy to 1.0. This will give your scene a flat, shadowless light that suits a cartoony effect.

Create a Circle mesh object. TAB into Edit mode; all the vertices should be selected, so you can add a face joining them all just by pressing F. Next, select just one vertex with RMB. Go to the Mesh → Snap menu and select “Cursor to Selected”. TAB out of Edit mode, and go to the Object → Transform menu and select “Origin to 3D Cursor”. Putting the geometry origin at one edge of the circle, rather than its centre, makes it easier to apply the upcoming transformation via an array modifier.



TAB back into Edit mode. Select two vertices equidistant from the one you used to position the geometry origin. Turn on proportional editing (Mesh → Proportional Editing → Connected). Now press S to scale the two selected vertices closer to each other, and use the mouse

wheel to regulate the proportional-editing falloff to get a reasonable leaf or petal shape, as at right.

Once you've got a decent shape, select all the vertices and rotate them by a small angle, say, 5°, e.g. R Y 5KEY ENTER. This will keep the overlapping copies of the object from running into each other and producing some unpleasant rendering effects.

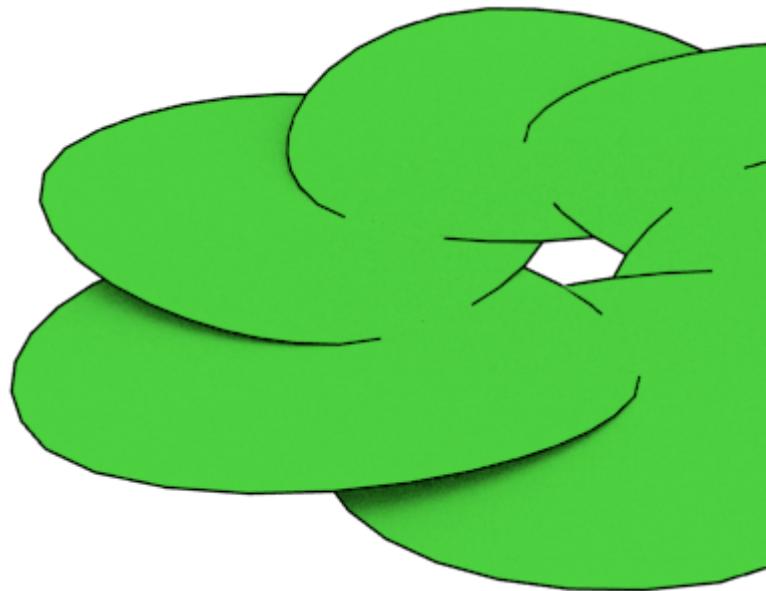
TAB back into Object mode. Now add an Empty at the current position of the 3D cursor. Rotate it about the Z-axis by 60°: R Z 6KEY 0KEY ENTER. Since this shares the same geometry origin as your circle, it can be used to apply a rotation to the latter without it turning into a spiral.

Select your circle again. Add an Array modifier: uncheck Relative Offset (and ensure Constant Offset is off as well), and turn on Object Offset. Click on the field for entering the object name, and in the popup menu that appears, select the name of your Empty (it should be called “Empty”). Set the Count to 6. You should now see a leaf-like or flower-like arrangement of 6 copies of your circle. While you're at it, give it a suitably plant-like colour, like leaf green.

Now go to the Render Context . Look for the “Freestyle” checkbox near the bottom, and check it.

Switch to the Render Layers Context , and you should see that a “Freestyle” panel has appeared. Click the “+” button next to the “Freestyle Line Set” list to create a new entry. You can leave all the settings for the new Line Set at their defaults.

Now hit F12 to render, and you should end up with something like this:



4.32.2 See Also

- Video tour of just about every Freestyle setting.

4.33 Overview

This section will show you the Animation system as it is in Blender 3D. Most of the features will be explained and some tutorials will follow. It is assumed that the user has a good understanding of Blender here.

4.33.1 Index

- **Advanced Animation**

- Introduction
- Guided tour:
 - Armature Object
 - Armature Object in Object Mode
 - Armature Object in Edit Mode
 - Armature Object in Pose mode
 - Mesh Object
 - Connection between Armature and Mesh
 - Envelope
 - Vertex Groups & Weight Paint
 - Shape Key
 - Constraints
 - Copy Location
 - Copy Rotation
 - Track-To
 - Floor
 - Locked Track
 - Follow Path
 - Stretch-To
 - IK Solver
 - Timeline Window
 - Graph Editor
 - Dope Sheet
 - NLA Editor
 - Introduction To NLA Editor
 - The Stride feature
- Working example: Bob
 - Build The Rig
 - Deform The Mesh
 - Create A Walk Cycle

4.34 Introduction

Welcome to the wonderful yet complex world of computer animation! Through these pages I will try to show you everything old and new about the new animation system in Blender 2.4. But, before we get started, there are some basic notions about datablocks you should know. Animation in Blender is based on the fact that you have something moving in a Blender scene. For example, a ball bouncing on a floor plane:

-So you have a scene datablock, which holds some info about the scene itself, as you can see in the Render button window (F10KEY). -You populate this scene with various objects (which in this case refers to containers for data, not the actual mesh data that shapes the object itself). The only goal of an object is to hold the whereabouts of the data you want to see in your scene. It also holds the object *instance's* properties such as "does it have soft body or particle options, and do we draw its name?". Most of the info on an object can be seen in the Object Window (F7KEY).

An object links to all of the data you can see in a 3D view such as **mesh**, **curves**, **nurbs**, **lattices**, **armatures**, **metadata**, the **empty** property, text, camera and lamps.

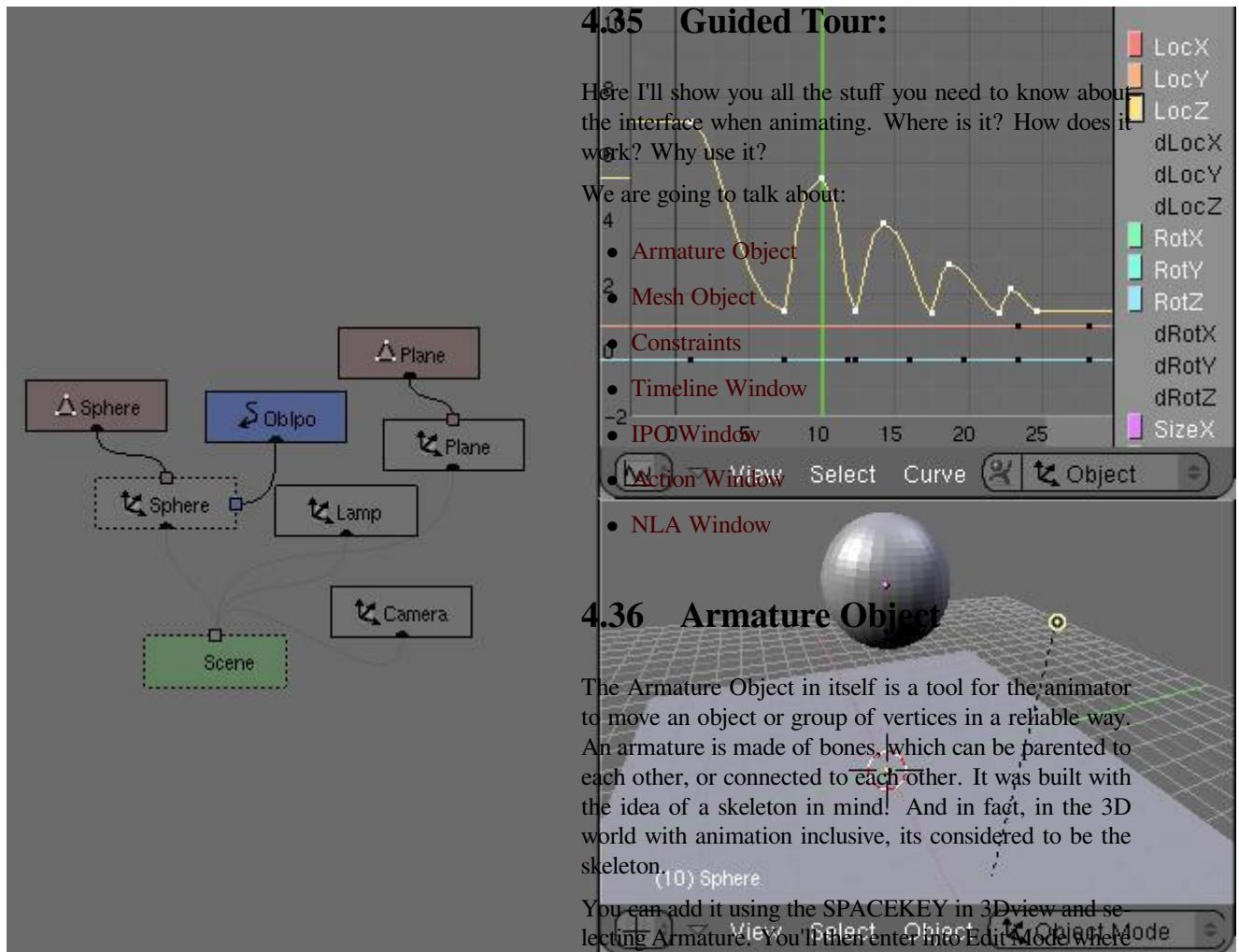
So the ball you just added to the scene is in fact a mesh, linked to an object that is in turn linked to the current scene.

Now there are also data blocks you can't see in 3D view, such as **material**, **texture**, **Ipo**, **action** and **image**. Instead, you have a special window in which to edit them. This is the idea behind the Blender interface, wherein each data block has a window for you to edit the data.

So back to this bouncing ball: It's also moving across the plane! So an ""Ipo"" data block is linked to the object, telling it where in space the object will be at each frame of the animation. This Ipo is editable in the Ipo window when selecting the ball in 3D view. In Blender, the work you are performing is always on the currently active (selected) object and data.

Note: In Blender 2.5x and later the OOPS view has been removed.

Looking at the OOPS (object oriented programming system) view (or SHIFT-F9KEY), we can get a good idea of the internal data structure:



Again, you are working in the scene “Scene”, with an object “Sphere” linked to the mesh data block “Sphere” and the Ipo datablock “ObIpo”. Why is that important? Because from there, you can start playing with the datablocks, linking them all around your projects to reuse old work. For example you can create more than one Ipo, and use the one you want, or tell more than one object to use the same Ipo, or to use the same object in more than one Scene.

Most of the linking job can be done in the Edit button window (F9KEY). Where you can tell an object to use another mesh’s data block for Ipo, material, texture or image. There is always a little dropdown menu button for you to select an already-existing data block.

Now, when it comes to animation, you have to understand the way Blender handles data very well, because using Blender is always a matter of plugging data blocks together when working with Ipos, actions and NLA objects.

4.35 | Guided Tour:

Here I'll show you all the stuff you need to know about the interface when animating. Where is it? How does it work? Why use it?

We are going to talk about:

- 4 • Armature Object
- 2 • Mesh Object
- 6 • Constraints
- Timeline Window
- IPOWindow
- Action Window
- NLA Window

4.36 | Armature Object

The Armature Object in itself is a tool for the animator to move an object or group of vertices in a reliable way. An armature is made of bones, which can be parented to each other, or connected to each other. It was built with the idea of a skeleton in mind! And in fact, in the 3D world with animation inclusive, its considered to be the skeleton.

(10) Sphere
You can add it using the SPACEKEY in 3Dview and selecting Armature. You'll then enter into Edit Mode where

you can add or move bones to build your default rig. An armature has 3 states. You can switch using the dropdown menu in the header of the 3Dview or use the TABKEY to switch between Editmode <-> [Objectmode/Posemode] and CTRL-TABKEY to switch between Objectmode <-> Posemode:

- **Object Mode:** Your armature is like any other Object, you can move it around the scene, scale it, rotate it and edit options in the button window.
 - **Edit Mode:** Your armature is in what we call rest position, you can modify the bones it contains.
 - **Pose Mode:** Your armature is ready to be animated, each bone can be moved, scaled or rotated, constraints get applied, you can pose your character and animate the bones’ behavior over time.
- Take note that Pose mode is now a state of the armature you can switch on/off using CTRL-TABKEY. So when in Pose, you are still in object mode (you can select another object, contrary to the edit-mode)

Note: The following 3 pages of this tutorial contain screenshots and discuss techniques that are only available in Blender 2.40a and later. Refer to the [Blender 2.40a release notes](#) on Armature draw types and Armature envelopes.

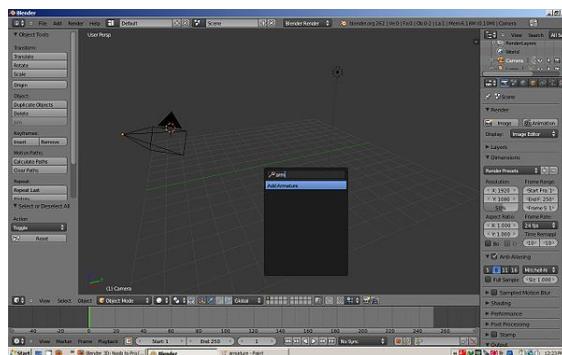
4.37 Armature Object in Object Mode

4.37.1 The Armature Object

Armature Object is like any other object type:

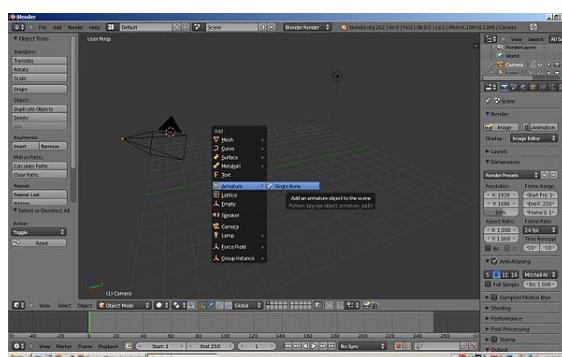
- It has a center, a position, a rotation and a scale factor.
- It can be edited.
- It can be linked to other scenes, and the same armature data can be reused on multiple objects.
- All animation you do in object mode is only working on the object, not the armature's contents like bones.

Try it now: add an armature to your scene: SPACEKEY --> Add --> Armature.



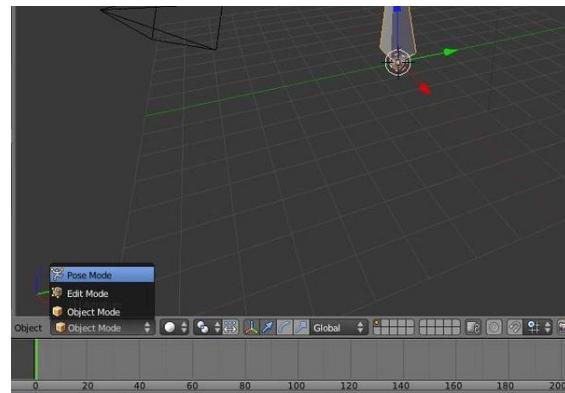
How to add an armature bone in Blender 2.62

Or you can press the Shift + A keys, and select single bone from the armature menu.



How to add an armature bone in Blender 2.62

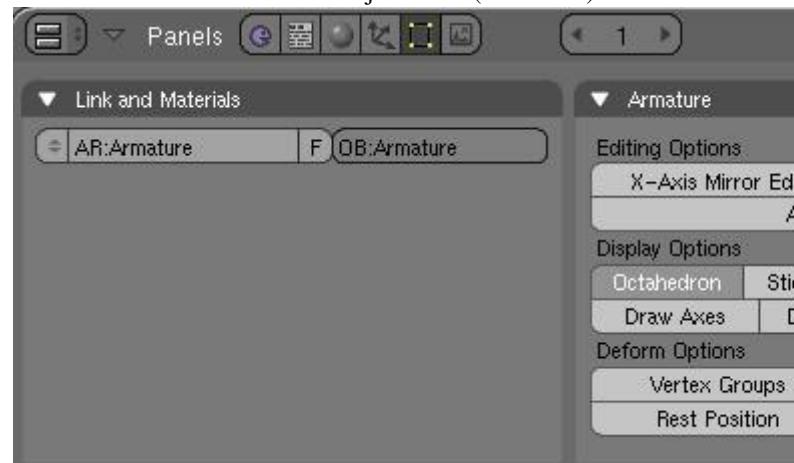
When you add a new armature, you'll enter editmode automatically. To switch between modes, use the TABKEY or the dropdown menu in the Header of the 3Dview window (This might not be the case with blender 2.49b, you won't enter edit mode in that version):



Modes that relate to Armature bones

4.37.2 The Edit Panel When in Object Mode

This is how the edit panel looks after you have added a new armature and switched to object mode (TABKEY):



- Link and Materials panel:

- The AR: field let you rename your armature Datablock. The dropdown is a quick way to select which Armature datablock you want to connect to this armature. You can keep more than one version for the same character. Useful when you have a special move to achieve in a shot, you can turn on an armature for a special purpose.

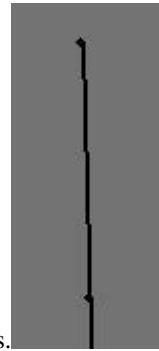
- The F button is an option to assign a Fake user to the Armature. Again if you have more than one armature for your character, it's a good idea to turn the Fake on, because if your armature datablock is not used (linked) it's not going to be saved in your .blend files. You can always do batch Fake-assignment of armatures by opening the Datablock browser (SHIFT-F4KEY), go in Armature datablock, select all the armatures you want to keep, and Press the FKEY.

- The OB: field is just to Rename your armature Object to something more cool and useful than Armature... Armature.001...

- Armature panel:

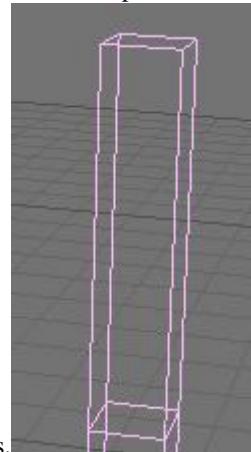
- Editing Options:

- X-Axis Mirror Edit: Not really useful now, it's more of an editmode option. This feature tells Blender you want to replicate all of your bones on one part of the Armature to the other. It's a clean way to just do half the job ;). The axis of mirroring is X so left<->right in frontview (NUMPAD_1KEY) and the center is the center of the armature object. We will see this feature in detail in the next page.
- X-Ray: This option will let you see the armature through anything in the scene, solid or not. It's useful to see where your bones are in your character so you can select them.
- Automatic IK is a Posemode option. It lets you pose a chain of bones as if the bone you were holding was an ik target. More info in Posemode page.
- Display Options: These options give you the chance to visualise your bones in various ways. Also note that there is some specific options and features regarding the display mode you're in.
 - Octahedron: This is the default view. Nothing exciting except you have a good idea of the rolling of the bones.
 - Stick: This display mode is really useful when you have a lot of bones in your view. It lets you “unclutter” the screen a bit. It draws the bones as tiny sticks.



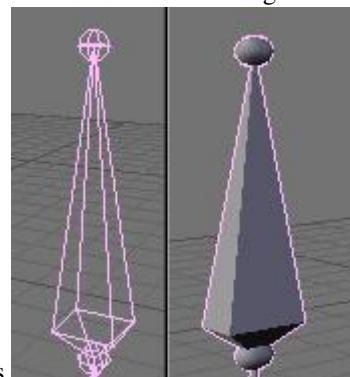
sticks.

- B-Bones: It's more a feature than a display mode. This is only useful to visualise the effect you get when you activate the B-bones (Bézier-Bones). Each bone acts like a curve handle and lets you get extremely curvy poses. This will be exposed in the following pages.

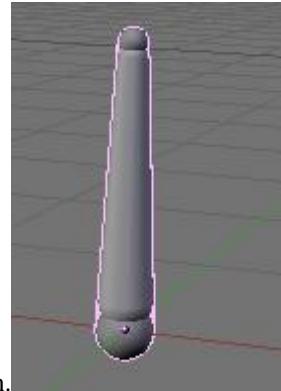


pages.

- Envelope: Again it's more a feature than a display mode. But in this case the visualisation will be useful to tweak your rig later. Envelope lets you easily tell which part of your character this bone will animate and it's visually possible to change the zone of influence exclusively in this display mode. The zone is only visible in Editmode or Posemode



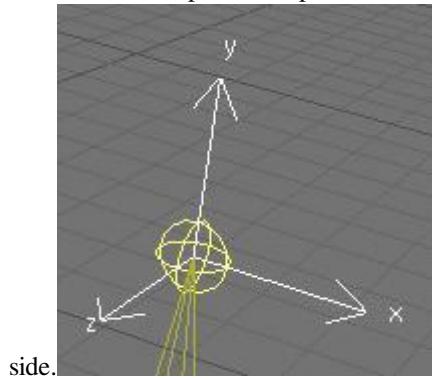
bones.



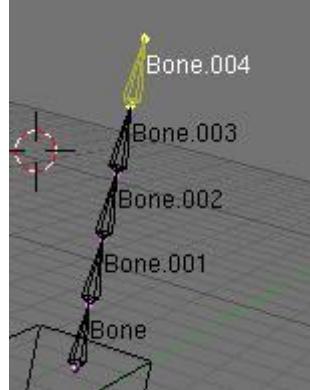
though.

- Draw Axes: To draw the axes on each bone of the armature when you are in Editmode or Posemode. Useful when you want to know where you are and

which axis to use in a constraint for example. Mental note: Y is up, Z is depth and X is side, contrary to object for which Z is up, Y is depth and X is



- Draw names: This lets you see names of bones whatever the mode you are in. It's useful again to edit your armature, create parent dependencies or add constraints.



- Ghost: This option lets you see a ghost of the armature in frames behind and over the current time. This is only working when you have an action linked to the armature, as we will see later.
- Step: This option lets you choose the frames interval between ghost instances.
- Deform options:
 - Vertex Groups & Envelope: These two toggles let you choose if you want the armature to deform your character using the Vertex Groups and/or the Envelopes. We will see that later.
 - Rest position: This will bring the character back to factory default (item as Edit-mode), and no actions will be applied to the armature so you can easily edit it in the middle of an animation.
 - Delay Deform: This was useful before as the old system was *very* slow. What it does is when you do a manipulation to the rig, it waits until you finish to update the view. Can still be useful though.

4.37.3 Extra Practice

This YouTube tutorial might also help: Link The short tutorial might be a help [Link 2](#)

4.38 Armature Object in Edit Mode

Now you've got your armature, but it's not much use until you add some more bones to it. Think about your body for a moment -- you've got this thing you call a 'skeleton', which for our purposes corresponds more or less to an armature object. Your skeleton consists of a number of bones (about 206, to be precise), but generally these are not independent from each other. If you move your femur (the bit of your leg between your pelvis and your knee) then conveniently the rest of your leg moves with it. In that example, the tibia/fibula would probably be counted as one bone, with the femur as their 'parent' bone. In this way, you build up a hierarchy of bones, making animation much simpler.

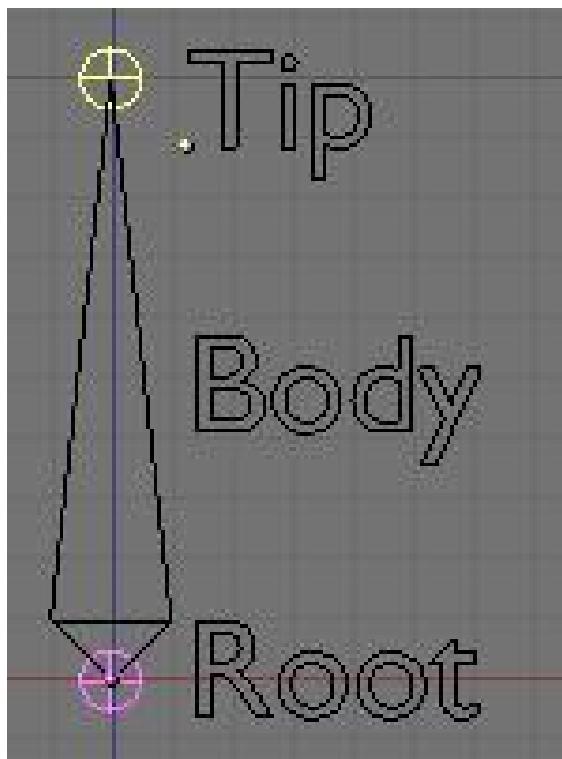
Editing an Armature Object gives you the chance to add, move or connect bones together. Whilst in edit mode, you will see all of the bones within the currently selected Armature.

When you create a new armature in Object mode a single bone will automatically be created for you, centered at the cursor position. Blender will then switch straight to Edit mode to allow you to add further bones. At this point we're just defining the default 'rest' position of the bones and specifying how they connect together -- you'll have to wait until [the next chapter](#) to find out how to create specific poses.

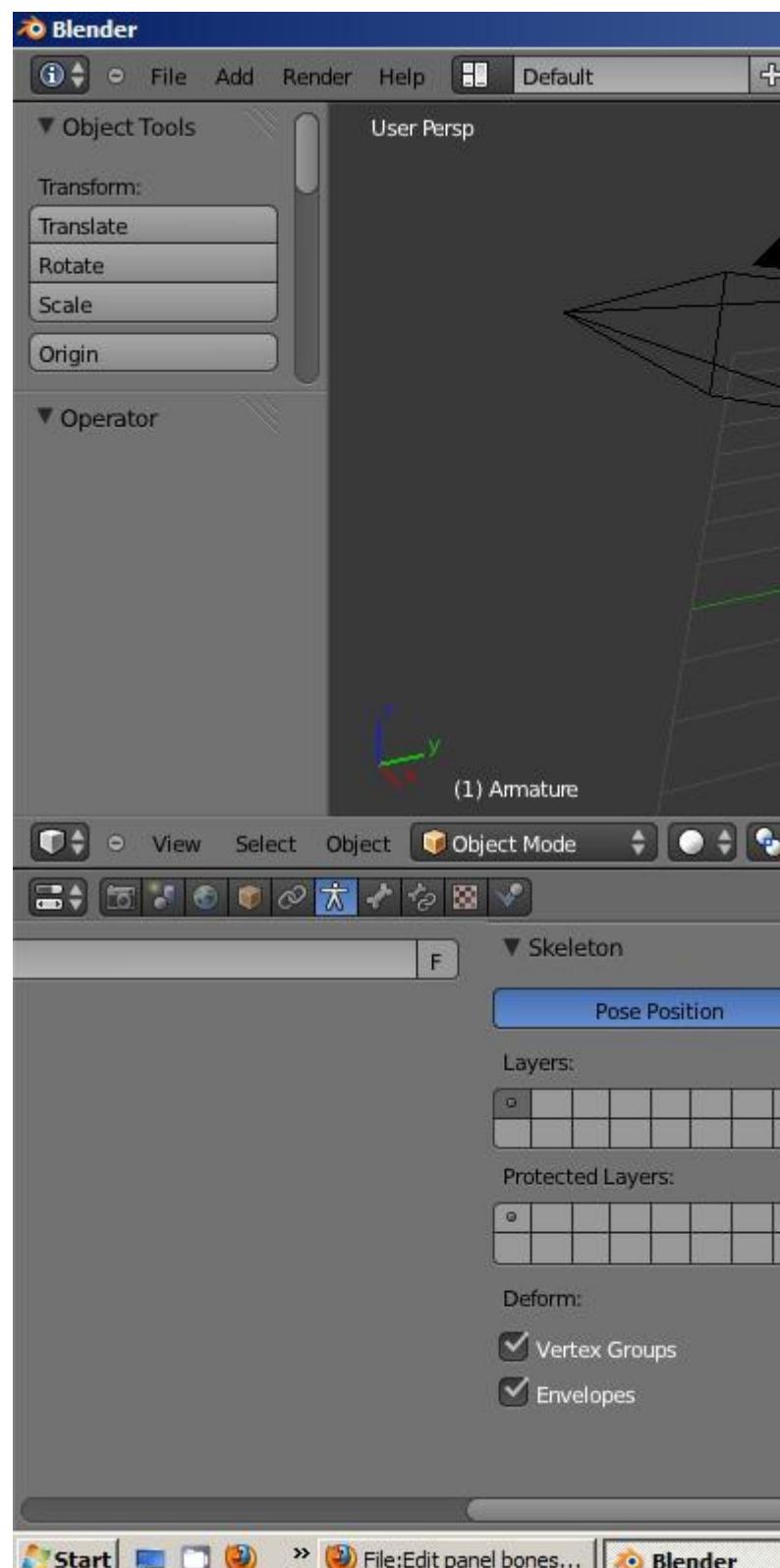
4.38.1 Now the basics about bones

Having created and selected an armature in Object mode, you can add and modify the bones in this armature by switching to Edit mode.

- You can add a new bone at cursor position by pressing SPACEKEY in the 3DView --> Add --> Armature.
- A bone has two ends: a root (the lower part) and a tip (the upper part). You can select and move the tip or the root independently with RMB, or you can select the entire bone by clicking on its body.
- You can extrude a new bone from the selection using EKEY. This will create a bone connected to the original one, meaning the Root of the new bone will follow the Tip of the original one. You can also CTRL-LMB to extrude a new bone. It will extrude to where you clicked.



4.38.2 The edit panel

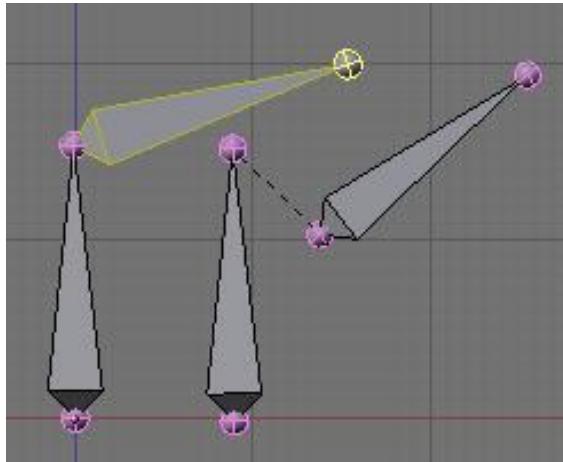


- Alternatively, you can connect two existing bones by selecting them one after the other and pressing CTRL-PKEY. You can then choose either 'Connected' (the child bone - the one you selected first - will automatically be moved so that it touches the parent) or 'Keep offset'.
- You can use SHIFT-DKEY to duplicate a bone
- Using the WKEY menu, You can subdivide your bone or flip the name of the bone between Left-Right (See Naming convention below).
- You can delete the bone with XKEY
- You can select a chain of bones (connected together) using LKEY, when you hover your mouse over a bone.

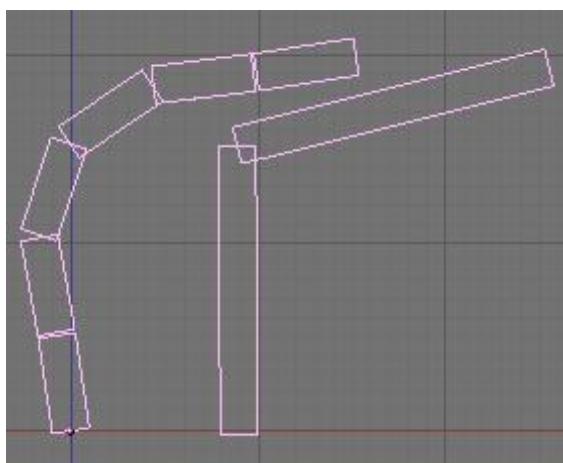
• Armature Bones Panel

- OB: this field lets you rename your bone.
- “Child of” Dropdown: lets you choose which bone will be the parent of this bone. If a parent

is selected, there will be a small button labelled "con", meaning connected. Setting the parent defines the relationship between your bones. When one bone has another as its parent, it will do everything the parent does, such as rotate, move and scale. A dotted line between the parent and child will appear. If you select Connected, the Root of the Children will go stick to the tip of the parent, giving you a chain of bones like the 2 bones in your arm.

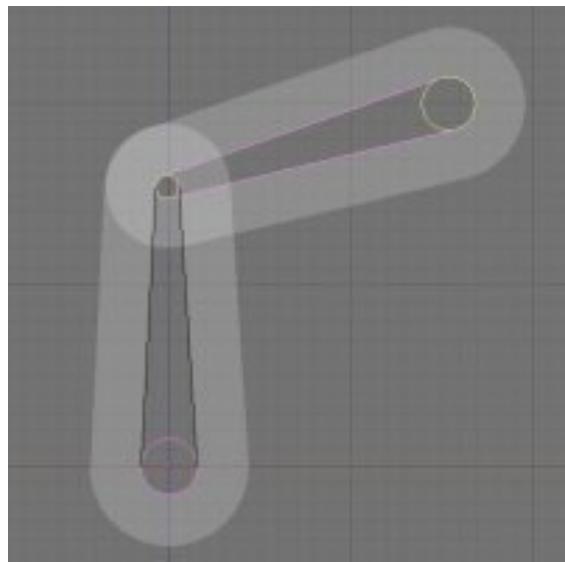


- Segm: If you set this value to something greater than 1, it will cut your bone into several little segments and deform them on a bezier curve - referred to as a 'B-Bone'. You need to create a chain of bones to really show off this feature though. In the example below, the image on the right has 1 segment, and the one on the left has 3 segments each (these are shown in Object mode to show the effect more clearly):

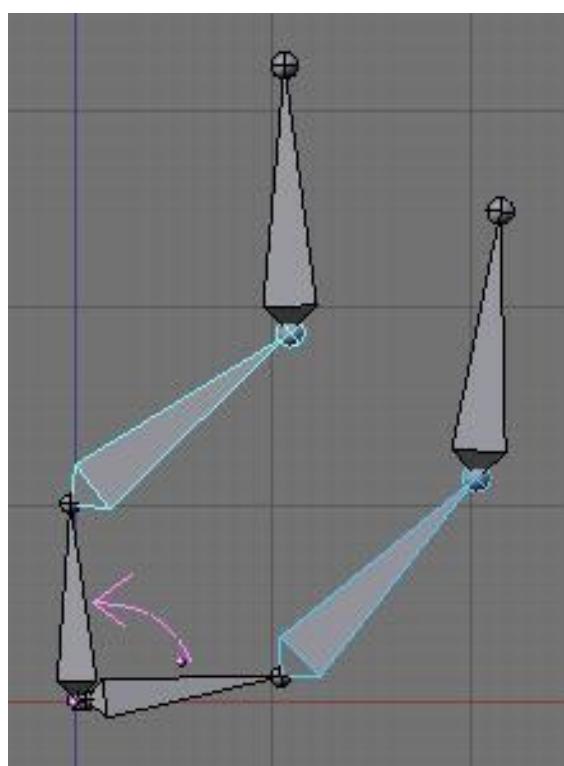
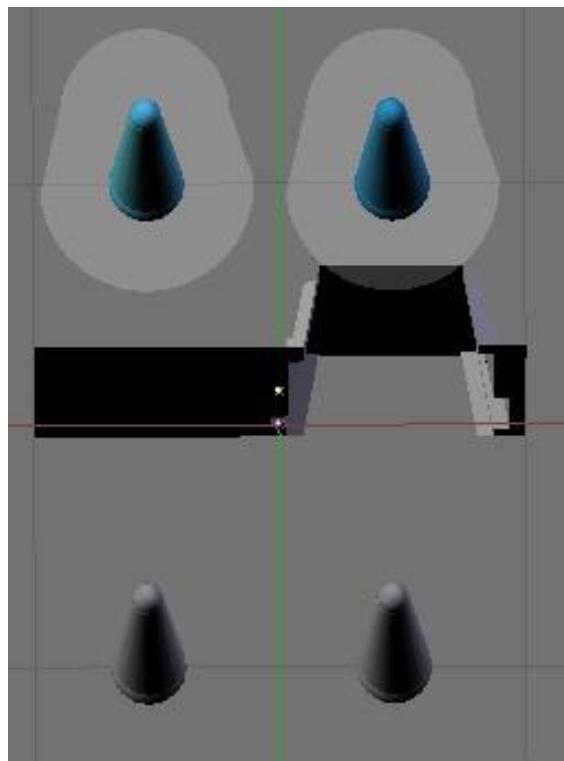


- Dist: This is the area of influence of the bone. It can be visualised using the **Envelope** display mode. We generally don't touch this field as there is an easier and faster way to change this

option. Turn Envelope on and select a bone. Then using ALT-S, you can scale the zone of influence. This has the advantage that you can do it on multiple bones simultaneously, and it works in both editmode and posemode:



- Weight: This specifies how strongly this bone will influence the geometry around it, relative to the other bones. If two bones crossing each other, both with envelope influence, have the same weight (like 1:1) they will influence the surrounding geometry equally. But if you set one to 0.5, the geometry will be affected more significantly by the other one, with weight 1. For example, in this image, 2 bones using envelope influence try to move the same geometry. The 2 on the left have the same weight, you can see the geometry didn't move. On the right, one of the bones has 0.5 so the bone with weight 1 is winning the tug-of-war!:
- Hinge: This tells the bone to remain motionless in a chain. It doesn't copy the rotation and scale of the parent. Useful for mechanical rig I would say, as you can animate the rotation of the hinge bone without having to correct it because the parent rotated:
- Deform: This lets you say if you want the bone to deform the geometry at all. Switching it off is like setting the weight to 0, except it's faster this way. Useful when using a bone as a target or a controller, i.e. a bone you just want to use to control other bones, but not the geometry itself.
- Mult: to deform geometry you can use **vertex group** and/or **Envelope**. The ability to mix both of these methods is handy for using one to tweak the other. For example, you might use envelope everywhere but tweak difficult places



manually with vertex group. We'll discuss this in more detail later on.

- Hide: This option lets you hide the bone. You can use it to hide the less important bones when you want to see what you're doing or for when you come to animate later on. For example, when you animate you don't need to see

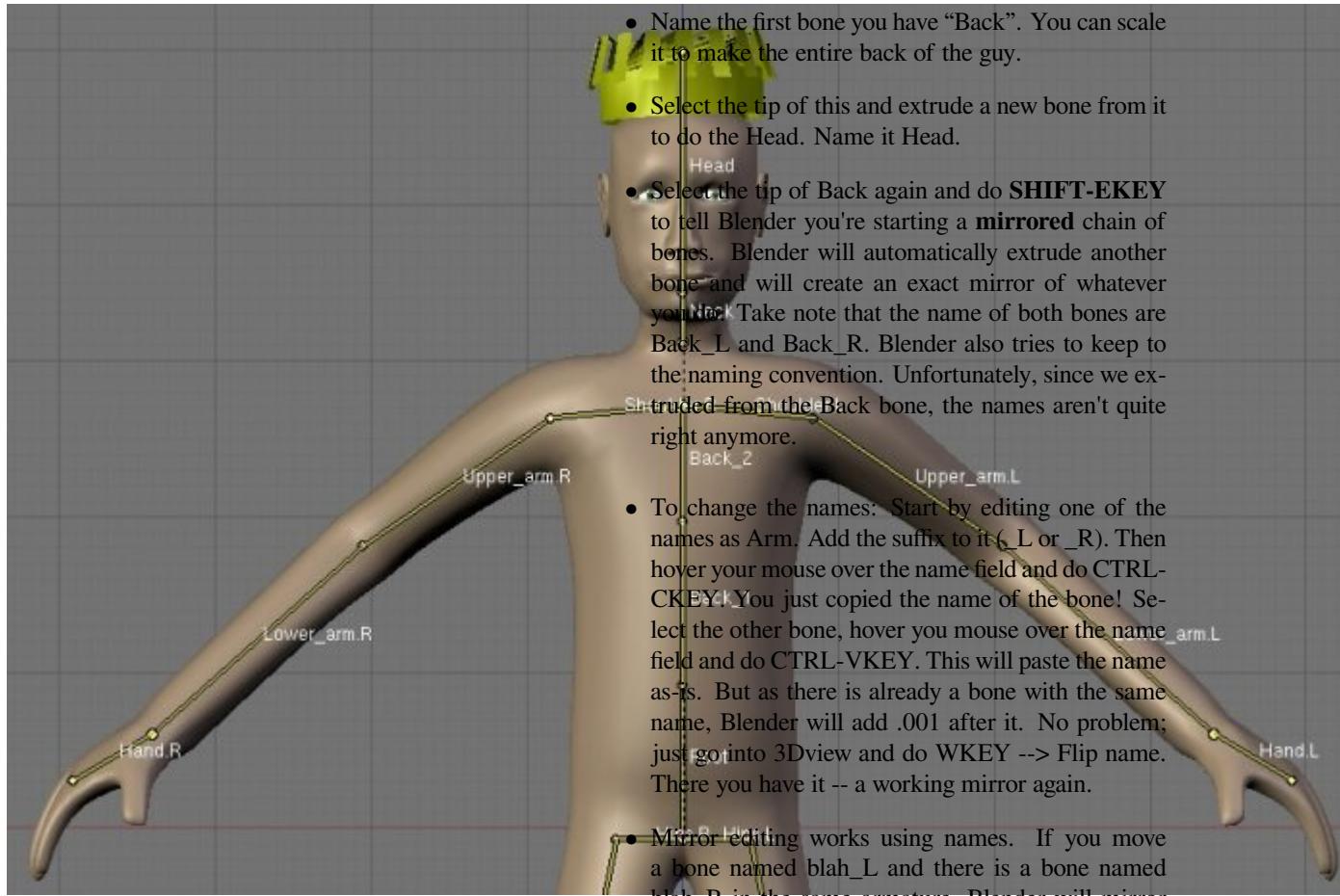
the entire chain of the leg, just the controllers. The values you select here apply to both Edit-mode and Posemode.

4.38.3 Naming convention

In many cases, rigs are symmetrical and can be mirrored in half. In these cases, it is helpful to use a left-right naming convention. This is not only useful for your own sake, but it gives Blender a hint that there is a pair of equivalent bones, and enables the use of some very cool tools that will save you some significant work.

- It's helpful to name your bones with something useful telling you what it's there for, such as leg, arm, finger, back, foot, etc.
- If you get a bone that has a copy on the other side, however, like the arm (you have 2 arms right?), then the convention is to call them arm.Left and arm.Right.
- Other alternatives are also possible, like _L, _LEFT, _left, .L, and .Left. Anyway, when you rig try to keep this left-right thing as accurate as possible; it will pay off later on.
- You can copy a bone named blah.L and flip it over using WKEY --> flip name. So the bone will be blah.L.001 after you copy it, and flipping the name will give you blah.R. Blender handily detects if the .001 version already exists, and increments the number for you.

This is an example of naming in a simple rig:



4.38.4 Mirror Editing

Now we come to the X-Axis Mirror Edit feature. This handy little number allows you to define only half of your character and tell Blender to automatically repeat the same actions on the other side. It's cool, it's simple and it saves a whole lot of time.

We will create a little guy out of sticks for the occasion -- don't worry about the geometry yet.

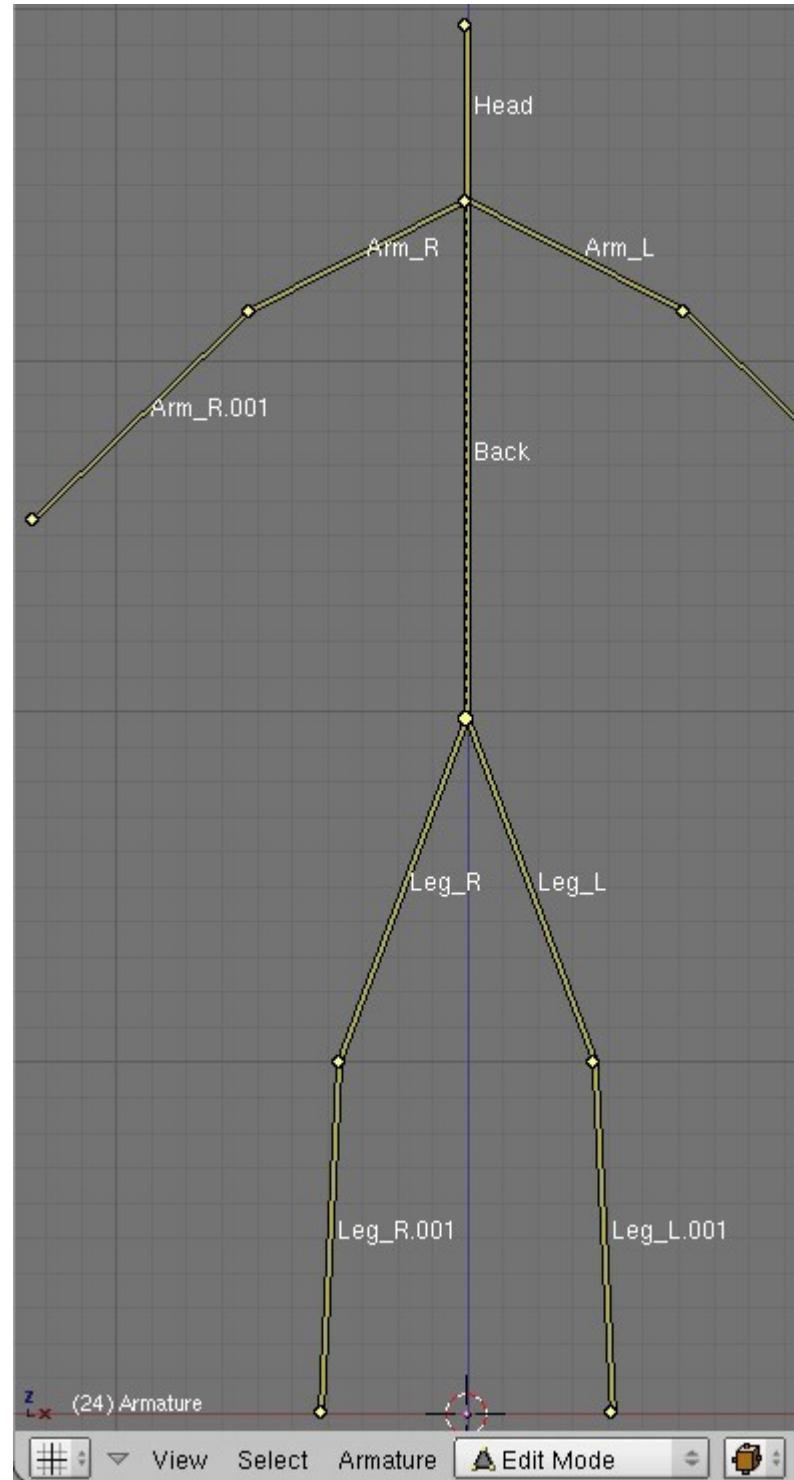
- Create a new, empty scene. Since the mirror editing feature mirrors along the X Axis, make sure you've got the front view selected (NUMPAD_1KEY) so that the X Axis runs from left to right. Add a new armature to the scene (SPACE, AddArmature). Enable 'Draw names' from the 'Display options' section of the Editbutton panel, so we can see what we're doing.
- Now enable mirror editing by pressing F9 on the buttons window and clicking the **X-Axis Mirror** button in the **Armature** panel under **Editing options**. You'll also need to use the center of the armature (indicated by a purple dot) as the center of your rig, otherwise the symmetry will go wrong when we come to create the mirror image.

- Name the first bone you have "Back". You can scale it to make the entire back of the guy.
- Select the tip of this and extrude a new bone from it to do the Head. Name it Head.
- Select the tip of Back again and do **SHIFT+EKEY** to tell Blender you're starting a **mirrored** chain of bones. Blender will automatically extrude another bone and will create an exact mirror of whatever you did. Take note that the name of both bones are **Back_L** and **Back_R**. Blender also tries to keep to the naming convention. Unfortunately, since we extruded from the Back bone, the names aren't quite right anymore.
- To change the names: Start by editing one of the names as Arm. Add the suffix to it (_L or _R). Then hover your mouse over the name field and do **CTRL-CKEY**. You just copied the name of the bone! Select the other bone, hover your mouse over the name field and do **CTRL-VKEY**. This will paste the name as-is. But as there is already a bone with the same name, Blender will add .001 after it. No problem; just go into 3Dview and do **WKEY --> Flip name**. There you have it -- a working mirror again.
- Mirror editing works using names. If you move a bone named **blah_L** and there is a bone named **blah_R** in the same armature, Blender will mirror the move you do to it, so make sure you follow name convention correctly.
- Then we can continue: extrude an other bone to make the lower part of the arm using **EKEY** or **CTRL-LMB**. The new set of bones should be **arm_L.001 arm_R.001**.
- Then we will add the legs. Up till now we have always worked from the tips of the bone. This is easy as Blender understands you want to create children of the selected bone, but to make the legs you need to extrude from the root of "Back". So go ahead, select the root of "Back" and do **SHIFT+EKEY** to start a pair of chains. Rename them to "leg"+suffix.
- Now take note that doing so will not parent or connect the new bones to anything. We don't want it to be connected to the tip of "Back", it would look silly. But we want it to follow the body!
- The way to go is to parent the two legs we just created to the "Back" bone. The old way (pre 2.40) was to select all bone and select the parent manually in the drop down. In the new version, editmode and posemode have an *active bone*. The active bone is the last bone you selected. In this case we can't work with more than 2 bones selected. Select the child (a leg) then select the parent (Back) and Do **CTRL-PKEY**. A menu will popup asking *Connected*

or *Keep offset*. For now use *Keep offset*. Do this for the other leg as well.

- it's also possible to remove parent easily. Select any bone you want to remove parent relation from and do ALT-PKEY. A menu will popup asking if you want to clear all or just to unconnect. Of course you don't need to select the parent and/or the child for this to work since any parent relationship will be cleared. So if you do that on a bone which is parent of 5 bones, then immediately all the children will be parentless.

- Extrude one more time to get a leg with 2 bones.



- Now you can go into Posemode and pose your guy as you want.
- You can move the entire guy just by moving the "Back" bone, since this is how we built him. This bone is the highest in the bone hierarchy, "The daddy of all bones", you could say!
- Turn on the Stick display mode and enjoy your guy made of sticks!

FASTER WAY TO NAME YOUR BONES 1) Create one side of the armature complete with correct bone names like the way this tutorial describes. 2) Copy the

side of the armature you just created and paste it on the opposite side to form a complete armature. 3) On the side that you just duplicated, your names would have numbers added to them. Now select each bone on 3D View and hit “W” then select “FLIP NAME”. MAKE SURE YOUR PARENTING IS THE SAME AS THIS TUTORIAL DESCRIBES!

4.39 Armature Object in Pose mode

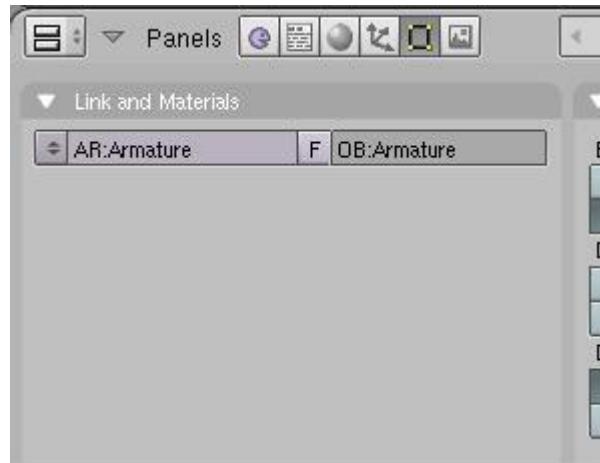
Posemode is a very versatile place where you Animate your character, create and manage constraints and apply your rig to your character.

Contrary to Editmode, Pose mode isn't an obligatory mode where you can't do anything else. It's now part of the UI like any other object. A good example of it is you can be in posemode and still select another object.

4.39.1 So What Can You Do?

When you are done building your armature, you can go into Posemode to add constraints and start creating actions. There are also some new tools accessible in Pose-mode that you may want to look at. You can easily get into “pose” mode by selecting the mode from IPO type list box in the left portion of the lower screen.

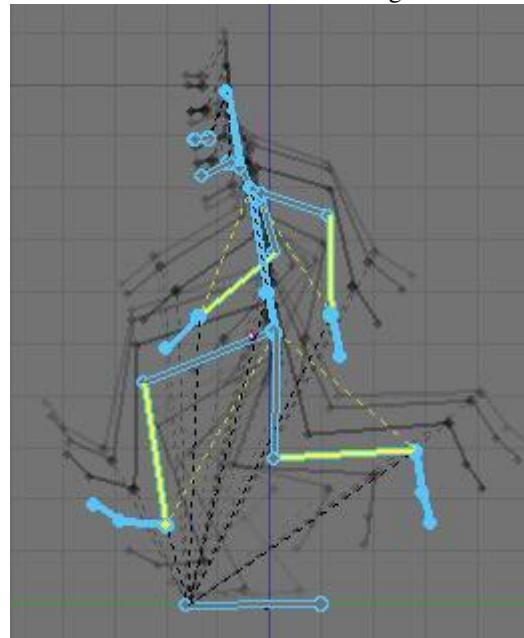
The panel has changed a bit too:



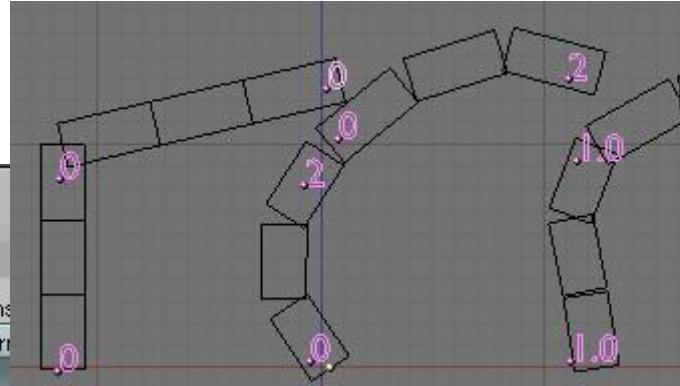
- What's new in the panels?:

- You can use the Automatic IK feature in the Editbutton(F9) to pose a chain of bones like it was an ik chain. Its usefulness is very limited though. It works well only if there is no other ik solver in the chain, and if your chain is isolated from the rest of the rig.
- Ghost: in the armature panel the ghost option lets you see the action linked to the armature

over time. Also called onion skinning.



- There are two number fields to better tweak the effect of B-Bones. The in/out is used to tell the scale of the virtual handle of the bezier curve. In is the Root of the bone, and Out is the Tip. The bigger the value, the bigger the effect of rotation.



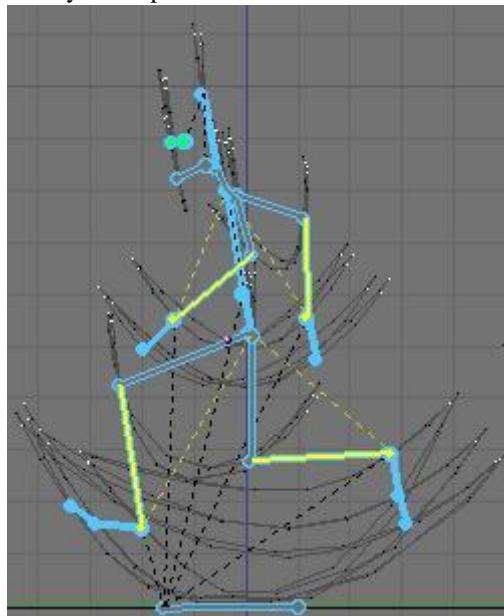
- There is now a Constraint panel where you can add a constraint to a bone, like any other object in the scene. This will be shown later.

- You can pose your rig using GKEY, SKEY and RKEY. Note that if the bone is part of a chain it can't be moved (except if it's the first of the chain, moving all the chain as they are all children), so you rotate the bone instead.
- You can do ALT-SKEY on one or more bones while in Envelope display mode to tweak the envelope size in real time while animating. Useful when for example you move the hand and some part of the character isn't in the influence zone; the result will be that some vertices will stay behind.
- You can do CTRL-CKEY to copy stuff from a bone to bones. The options are location, rotation, scale

and constraint. Constraint is very handy when you want to copy a constraint to other bone. The way it works is easy.

- The W-KEY menu gets some neat options too:

- Select constraint target: Will select the target of the bone's constraint currently selected.
- Flip name: Yep, you can flip name in Pose-mode too.
- Calculate/Clear path: This is a visual way to see the action linked to your armature. You can select just some bones and ask Blender to show you the paths of the bones.



- You can pose your character and select all bones you want to see included in the action and press I-KEY.

You can insert a key just for loc, rot or size. Avail will add a key to all available channels in IPO window (all channels you previously added something).

- When you insert key for your armature, a new action is created and linked to the armature if there was no action before. You can also see the curves of each selected bone of the armature in the IPO window. We will see action window and IPO window later.

- You can parent a bone to an external object by selecting this object then selecting the bone in question so it's active (The armature is in Posemode so you can select a bone). Do CTRL-PKEY. Then when you move the bone the object will follow. This kind of Hard relationship doesn't include any bending at all. It's useful when doing robot rigs as you are just moving objects around.

4.40 Mesh Object

This section will explain how to deform your mesh using the armature.

There are two ways to tell Blender which vertex will go with which bone: Vertex group, and Envelope.

There is also a tool useful when animating which is part of the mesh object: the Shape key, to create a preset deformation. For example: deform the face to look like a smile.

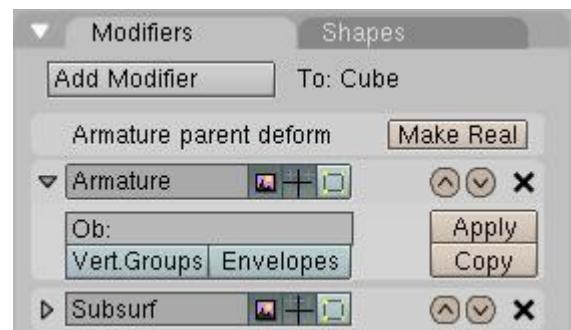
- Connection between Armature and Mesh
- Envelope
- Vertex Groups & Weight Paint
- Shape Key

4.41 Connection between Armature and Mesh

How to tell Blender: “use this armature to deform this mesh”

4.41.1 The Armature Modifier

Blender now has a Modifier stack (Editbutton, F9KEY). As such, we should use it over existing methods to pair mesh and armature, as the modifier stack is optimised and simple to use. Note: You don't need to parent the mesh to the Armature anymore. The only case you could need to do this would be animating the Armature object itself. Then the mesh should also follow the armature. In this case select mesh, then armature, and do CTRL-PKEY --> Object.



The clean way to do so is to go in the Editbutton window (F9KEY) and press “Add modifier” in the Modifier panel, then select “armature” in the dropdown menu. Then you'll get a new modifier “Armature” like the previous picture. Change the **OB:** field to the name of the armature object that should control the mesh. This step is very important! Without the armature object being defined, Blender won't know how to modify the mesh since

there may be multiple armatures within your world. To limit the effect of the modifier, you can enter the name of a vertex group in the **VGroup:** field. This will minimize unwanted distortions in very dense meshes. There are also fields to enable/disable the modifier when rendering, enable/disable when working to only move the armature (could get handy with massive character), and when editing (that's very handy, you can edit the topology while it's deformed). There are also two toggles to tell Blender what it should use to deform: Vertex Groups and/or Envelopes. You may have noticed these options are repeated also in the Editbutton --> Armature panel, but as the tooltip says: these two are used when you use virtual modifier (the old way) to keep compatibility with old files.

Parenting the mesh to the “armature” will create an old-way link, still visible in the modifier stack, but not very useful. The first entry with the “make real” button is what appends if you do a CTRL-PKEY to “armature”. You should not use that kind of connection when you see that. Press “make real” to get a working modifier.

4.41.2 The Old Way

This way is not recommended but can still be useful. When doing CTRL-PKEY to “armature”, you will get a menu like this:



- Don't Create Groups will just create a virtual modifier so you can deform the mesh (the “make real” button)
- Name Groups is almost useless now as blender will create a group for you when you do weight painting.
- Create From Closest Bones is a function to remember when you want to bake all your envelopes to vertex groups.

4.41.3 Tip: Bake envelope to vertex groups

The workflow is very simple. When you are done with the envelope’s tweaking and you have gotten the best out of it, delete the Armature modifier and parent the mesh to the armature. To parent it, go to object mode, first select the mesh and then the armature, then press CTRL-PKEY. Select *Create From Closest Bones*.

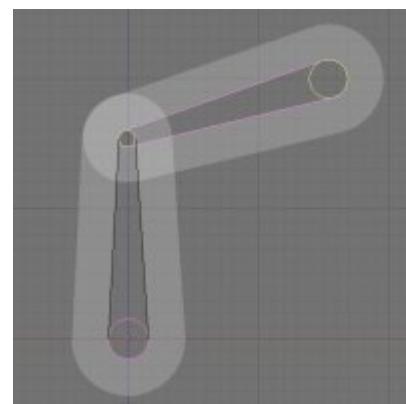
Do ALT-PKEY and redo the Armature modifier. Now all the envelope influence are converted to Vertex Groups.

This way you can further tweak influence zone using Weight paint. More info in the following pages.

4.42 Envelope

4.42.1 What is Envelope

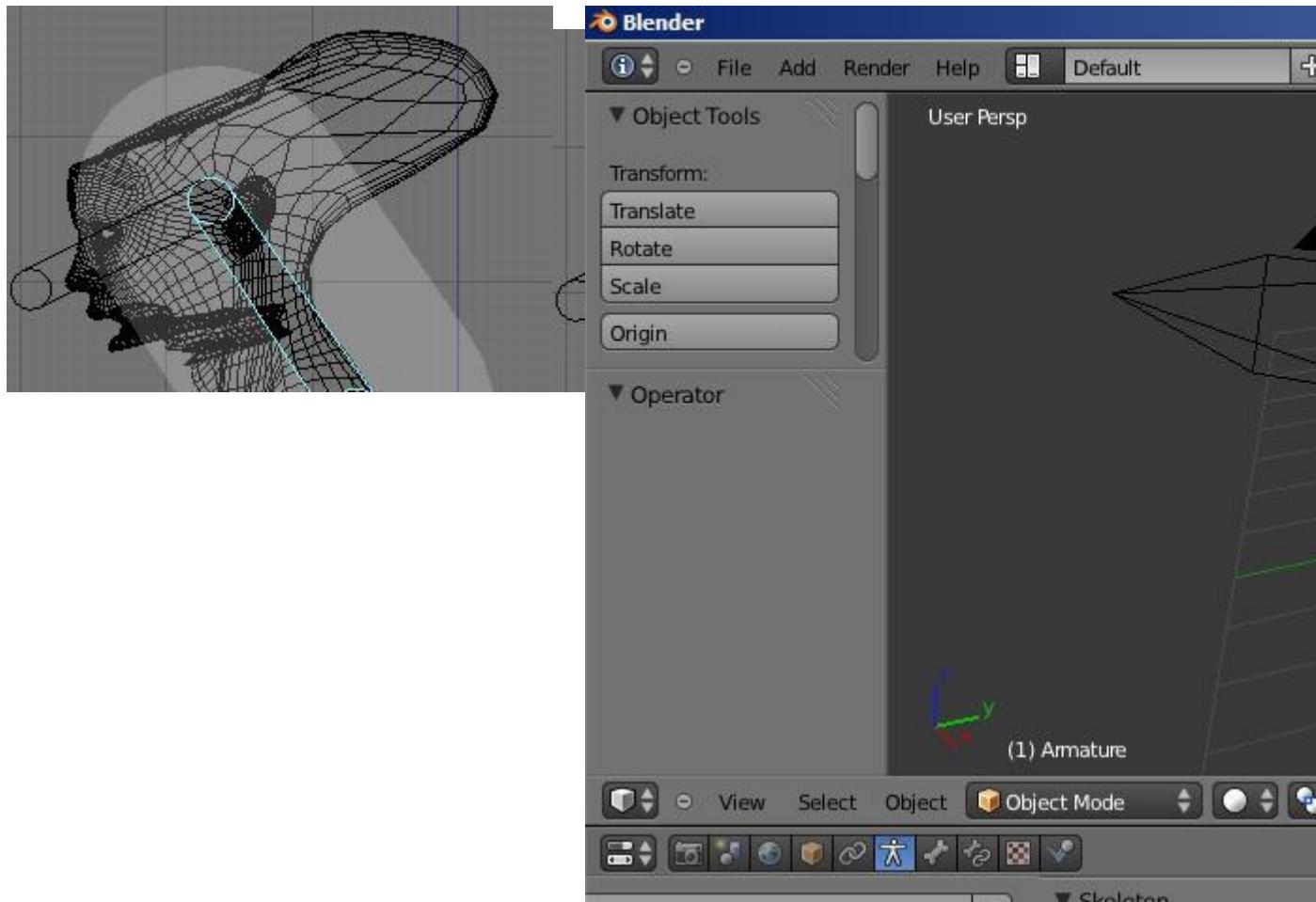
Envelope is a new visual tool to help you rig your characters faster and easier. It can often save you a lot of time. Each bone has a special area around it, allowing you to tell Blender what part of the geometry will follow each bone. This zone is customizable so you can move, scale and blend them together.



4.42.2 Edit Envelope

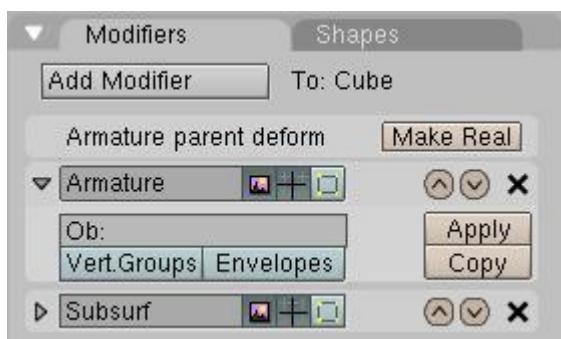
You can edit this white zone in Editmode or posemode by going in Envelope display mode (Buttons Window → Armature → Display Options → Envelope), selecting bones and using SKEY or CTRL-ALT-SKEY.

In Editmode: you can select the Tip, the Body or the Root and scale using SKEY. This area in the middle will assign a weight of 1 to all vertices contained in here. All vertices with a weight of 1 will completely follow this bone. The white transparent area around the center is a zone of influence which loses power as you go away from the center. This area is scaled when selecting the body of a bone and doing CTRL-ALT-SKEY. *In Posemode:* You can only scale the zone of influence with ALT-SKEY when in Envelope display mode. It's real time, and lets you tweak the influence while you animate. So if you notice there is a vertex not following in the new pose you just did: Just select the bone it should follow, and scale the zone a bit until the vertex goes back with his friends. Example:

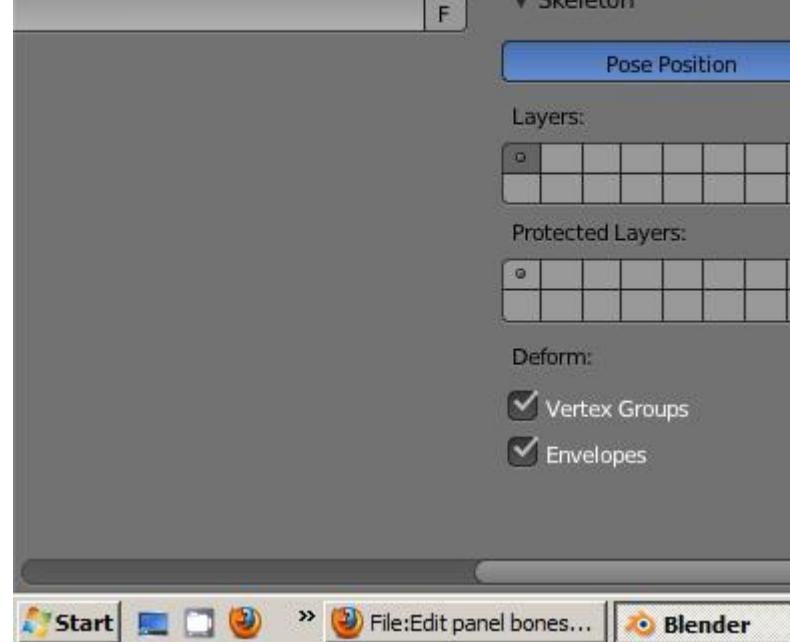


4.42.3 Envelope Options

It's possible to enable/disable the use of Envelope in the Modifier stack using the "Envelope" toggle.



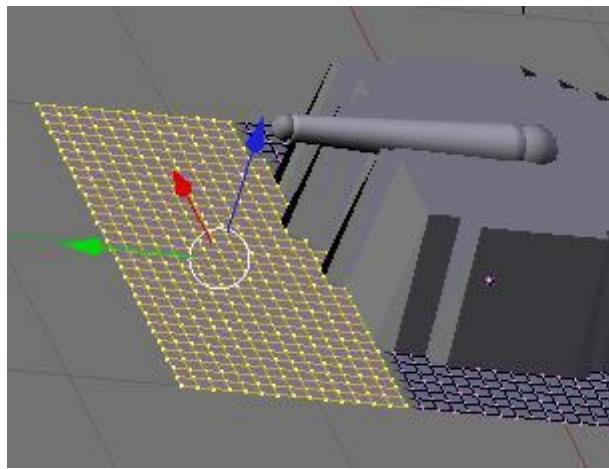
There are also two important buttons in the Armature Bones panel: Deform and Mult.



Enabling the Deform button will tell Blender to deform geometry with this bone. It's useful because in a more complex rig not all the bones are there to deform, some bones are just there to move other bones.

The Mult option will tell Blender to multiply the weight it gets from envelope (let's say 0.7) with the weight you painted in weight paint (let's say 0.5). The result will be

$0.5*0.7=0.35$ so in fact you just tweaked the envelope influence to 0.3 when it was at 0.7. If you don't want vertices to be part of the zone, you can always paint it with 0, as $0*(\text{something})$ will always give 0. This way you can give custom shape to your envelope. More on weight paint on next page.



In this example of a formerly flat surface you can see that all the selected vertices are not following the bone when it is moved upwards. This is because I painted a weight of 0 on them. In weight paint you'll see nothing. But just the fact that they are part of the group with a weight of 0 will make that possible. If Mult is off and you have both a vertex group and envelope, Blender will add value.

4.43 Vertex Groups & Weight Paint

4.43.1 What Are Vertex Groups?

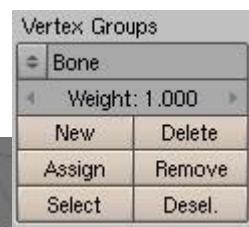
Vertex groups are very useful. You can use a vertex group to:

- Group vertices together while you model (keep a selection and come back to it later).
- Define which vertices softbody simulation affects.
- Define which vertices emit particles.
- Define which part of a mesh will follow a specific bone.

Vertex groups are specific to the Mesh object and can be modified in Editmode.

If you have vertices assigned to multiple groups (for example, in a character you may have some vertices in the “upper arm” vertex group that are also in the “lower arm” vertex group), you can assign weights to those vertices to specify how much relative influence the different groups have. A weight can range from 0 to 1 and is assigned

when you create the group. Let's take a peek at the GUI of vertex groups in the Editbutton(F9KEY):



From top down:

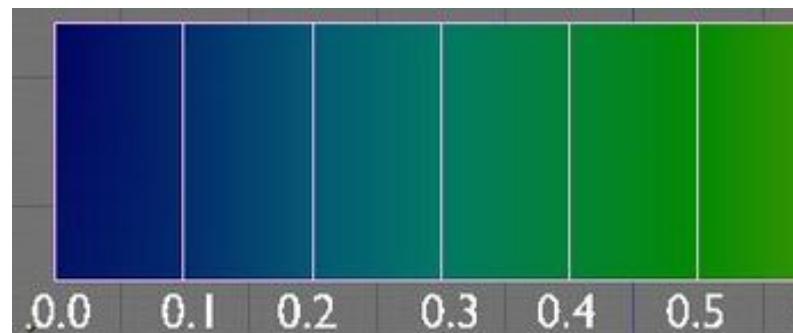
- The dropdown menu lets you select an existing vertex group or rename the current one.
- The weight numfield lets you choose the weight value assigned when you add vertices.
- You can add a new group or delete the current one.
- Assign or remove selected vertices to/from current group.
- Select/deselect all vertices in current group.

4.43.2 Weight Paint

As mentioned above, you may often find that you have some vertices that are assigned to more than one vertex group. By assigning weights, you can specify the relative influence each of the vertex groups have. You have two options to assign weights: 1) manually selecting each vertex and typing in a weight value, or 2) use weight painting to - you guessed it - paint weights.

Weight painting lets you paint weight values on the mesh like you were painting on a wall with a can of spray paint. It is a Mode of the 3Dview and is accessible in the 3Dview's header in the dropdown menu with Object-mode, Editmode and such. A hotkey is also available: CTRL-TABKEY.

In Weightpaint Mode, the first thing you'll notice is the blue color of the mesh. Blender provides an easy way to quickly visualise the weight value of each vertex. This is the color spectrum used:



When you are in Weightpaint mode you can paint all over the mesh as if it was a solid object on your desk. The paint

only works on vertices so don't try to paint in the middle of an edge or a face, it will never work. To help you in your task there is a new panel in Editbutton:



- The weight slider is just the same thing as the weight numfield we saw earlier in the vertex groups button. It's just easier to work with. It's simply the weight you want to apply to the vertices. In painting terms, think of this as the color.
- The buttons from 0 to 1 are shortcuts for weight value, to speed up the workflow.
- The opacity slider (and shortcuts) tell Blender what is the percent of the weight value you want to apply in one shot. If you set opacity and weight to 1 the vertex will turn red instantly. In painting terms, think of this as the pressure.
- “All faces” tells Blender if you want to paint on all faces in the mesh or just the visible one.
- “Vertex Dist” tells blender to use vertex distance instead of faces. When active, the painting will only check if the vertex is in the brush, then apply a weight value. If it's off, all vertices part of the faces in the brush will receive weight value. Turning on Vertex Dist can give good results when you have a lot of polys in your mesh.
- “Normals” will apply vertex normals before painting. This means Blender will take consideration of the direction the vertex is pointing when painting: the more it's facing away from view, the less it will receive value.
- “Spray” really makes it like spraying paint. Without it, a single click will only paint one value. With Spray on, each time you move the mouse a paint shot will be added. To get a good effect, use a small opacity value so the weight will increase slowly.
- “X-mirror” will tell Blender to apply the weight paint on the other group if there is one. Like Hand.L --> Hand.R. If you paint the group hand.L and there is a hand.R the paint will be copied over. For this to work your groups must be created, the name of the

groups have to follow name's convention (left right) and both side of the mesh need to be identical.

- “Wire toggle” toggles the visibility of wire while painting. Useful to find where the vertices are (activate the edit mode option “Draw all edges” to see even better).
- “Mix”/“Add”/“Sub”/“Mul”/“Filter” is how you want to apply the paint based on what is already there. Mixing will do a mean from brute weight value and current weight value, “Add”/“Sub” will directly add or subtract value, “Mul” will multiply (exponential painting) and “Filter” will paint based on alpha value.

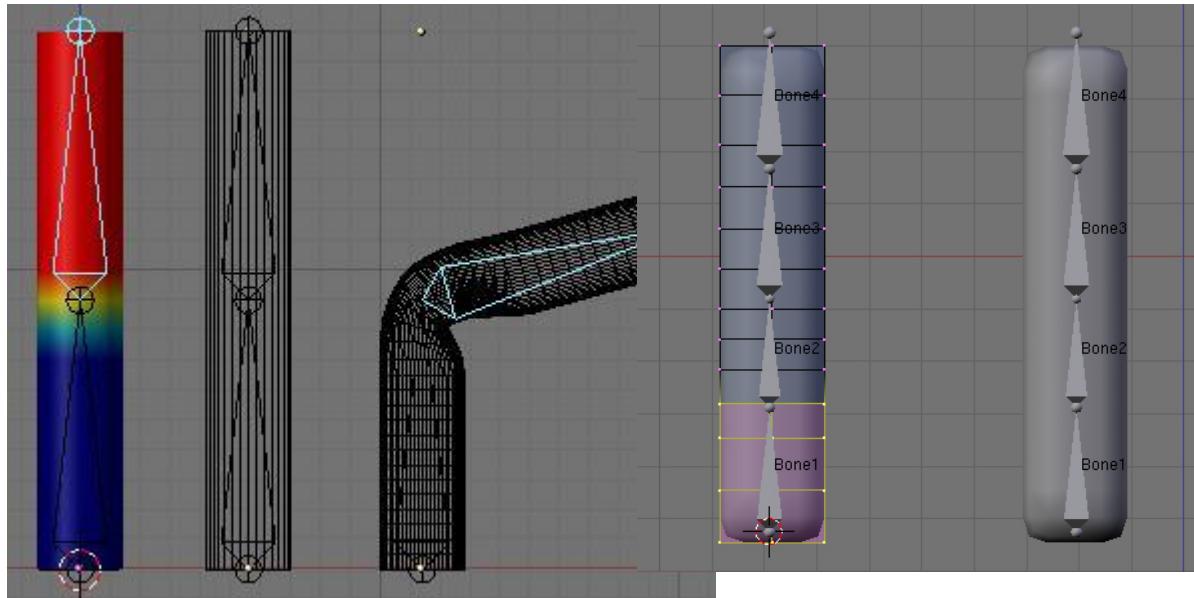
4.43.3 Vertex Groups and Armatures

So what use are vertex groups in rigging? You can specify what vertices will move when a bone moves. When you want to paint a mesh for an armature, do the following:

- Make sure the Mesh has an Armature modifier.
- Select the armature and enable Armature Posemode (CTRL-TABKEY).
- Select the mesh and enter Weightpaint mode (CTRL-TABKEY). (**Question: if we're in Pose Mode for the Armature, how are we supposed to select the mesh? Ans: right click on it.**)
- Select the bone you want to paint for with RMB.
- Paint the parts you want that bone to affect.

You'll notice that, if there is no group created when you first paint, Blender will create a group for you, and give it the same name as the selected bone. This is important, because when the “Vert. Groups” toggle is on in the Armature modifier, Blender will try to match bones with Vertex Groups based on the same names.

What happens when we try to blend groups together? See this simple example of 2 bones trying to bend a tube:



The Groups are painted so the body of each bone is in red and the zone between the two bones are gradually going from 1 to 0. This will bend nicely. If, for a special reason, you want a side to react differently, you can always move the bone while painting and try the new modification you just did. By the way, having Subsurf on while painting can be very cpu expensive. It's a good idea to turn it off.

4.43.4 Using Weight Painting with Armatures

Armatures are used for many purposes, but one common use is to deform a mesh with an armature. This example will demonstrate how weight painting can improve armature-deformed meshes.

In this example, we have two objects; each has an armature modifier applied. The one on the left is going to be the “before” and the one on the right will be the “after”.

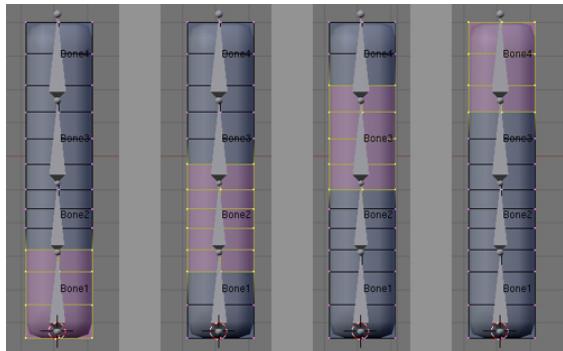
For the object on the left, take a look at the vertex groups as initially assigned (from left to right: Bone1, Bone2, Bone3, and Bone4). These same vertex groups were assigned for the object on the right:

Important: A bone in an armature will only act upon those vertices that are in a vertex group with exactly the same name as the bone.

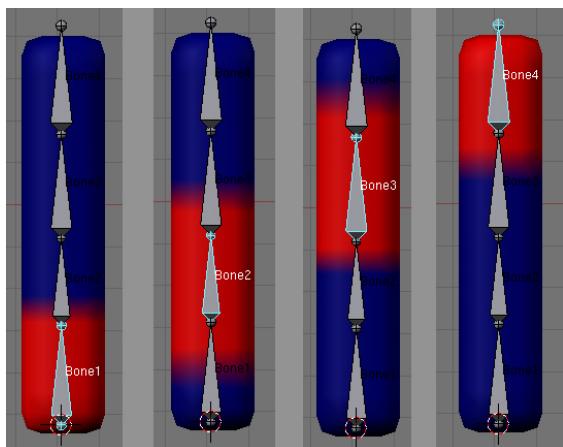
- In Blender 2.37 and previous, this was the ONLY way to get a bone to deform a mesh.
- In Blender 2.40 and on, selecting the “Envelope” button in the armature modifier will allow bones to deform even if you haven't assigned any vertex groups yet.

If you enter Weight Paint mode (CTRL-TAB with object selected) right after assigning the vertex groups, you can

The two objects in this example.



Vertex group assignments for each of the two objects.

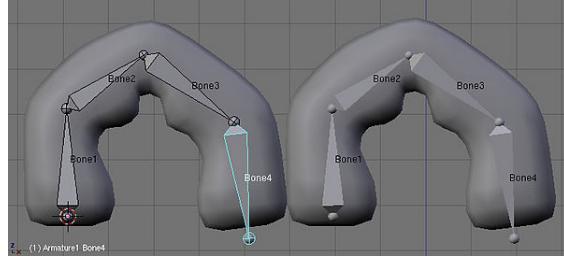


Initial weights for the vertex groups assigned above.

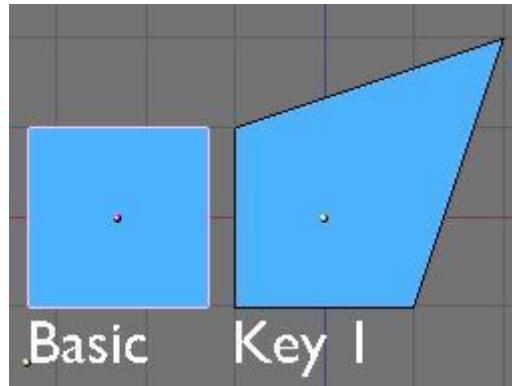
see that the vertex groups as assigned all have a weight of 1.0:

OK: both objects have vertex groups assigned and they have armature modifiers. Let's grab a bone (select the Armature, CTRL-TAB to enter Pose Mode, select Bone4, GKEY to grab, and move) to deform the mesh. We

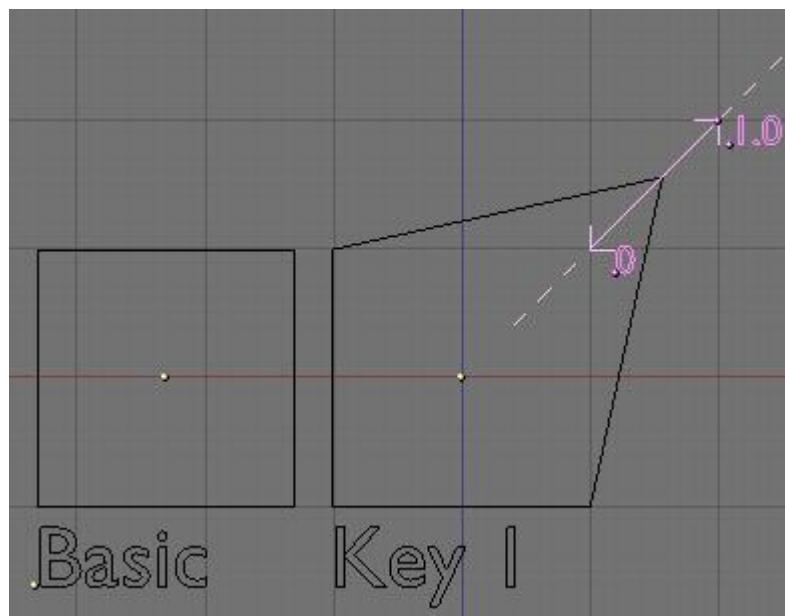
haven't made the objects different from each other, so after moving their armatures in the same way . . . there's still no difference. That's good.



Armatures deforming objects: before weight painting



When I play with the influence, this key has over the basic shape, the result will be as follows (0.5 in this example):



4.44 Shape Keys

4.44.1 Shape Key?

Shape keys are modifications of the original mesh that you can animate and mix with each other. Previously called Relative Vertex Keys (RVK), one of their major uses is to create facial expressions.

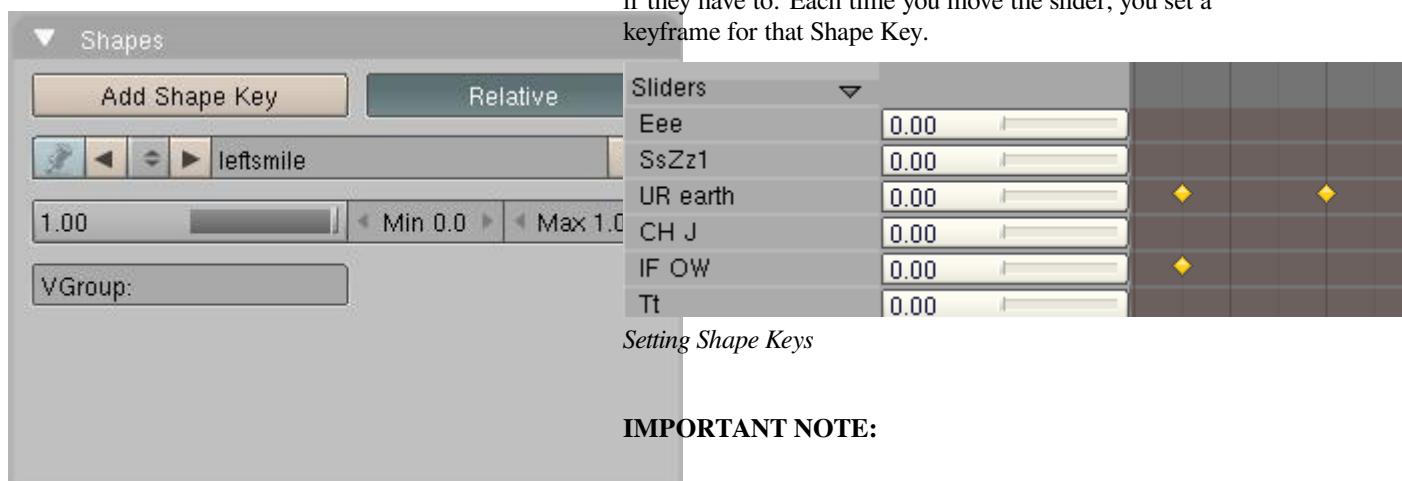


The idea behind Shape keys is simple. The basic, undeformed mesh you start with is the “**Basis**” shape. You can add a variety of different versions of this shape and store each one as a Shape Key for use in an animation (other software sometimes refers to these as “morph targets”). You can't add or delete vertices as Shape Keys only store the positions of vertices and not the creation/deletion of them.

Ok, to start out, take a plane. I'll add a new shape to it and move a vertex away:

If I play with the slider from 0 to 1, Blender will gradually apply the Key 1 shape with a constantly varying amount of deformation. So from 0 to 1, it will slide from the basis shape to the Key 1 shape. The movement of the vertices is linear from start position to end position and you can even set an influence greater than 1 or lower than 0 and Blender will extrapolate the movements of the vertices relative to the basis shape (marked as a dotted line above). This feature allows you to set general shapes such as a smile and then exaggerate the shape without needing to remake it.

4.44.2 The GUI



4.44.3 Shape Keys step-by-step

Here's a hand-held walk-through of how shape keys are used.

Start with the default cube or a more interesting mesh of your choice.

1: Select your object in **Object mode**. Go to **F9 Editing** window. Find and select the "Shapes" panel. Press the "Add Shape key" button. This adds a key called "**Basis**" and this stores the "*basic*" mesh in its undeformed state. Make sure the "**Relative**" button is pressed (should be default).

2: Press the "Add Shape key" button again. It now says "**Key 1**" and you have a slider and some other settings. Go into **Edit Mode**, grab a vertex and move it. **Exit Edit Mode**. The mesh returns to normal but you've just added a real Shape key and Blender has stored it.

3: Repeat step 2 for as many different shapes as you like. You can rename the keys to whatever you want. Normally you create each new shape by first selecting the "Basis" key but if you select an existing key from the list and immediately press "Add Shape Key" then enter Edit Mode, the mesh will already be deformed. This is useful for making similar but unique keys.

Using Shape Keys

1: Starting on frame 1, select each key one by one from the pop-up list (*or go to the Action Window, press the sliders button and select the Keys from the list there*) and click on the slider and move the slider forward then back to zero. Sometimes just clicking on the slider at the zero point is enough to set the key.

2: Move forward ten frames, select Key 1 from the list and move the slider to 1.00. You'll see your object deform. Move more frames and slide another key. And so on and so on. You can move them forwards and/or backwards as

you move through the frames and they'll just add together if they have to. Each time you move the slider, you set a keyframe for that Shape Key.

IMPORTANT NOTE:

You should add shape keys to a finished mesh. Don't work on a mirrored mesh or a partially finished model. Adding geometry (vertices, faces, loops, joining etc...) can result in the loss of the shape keys or to unpredictable results. Not always, but probably when you least expect it. There are scripts available to make some of these things possible.

That's the basics. Note that shapes will constantly transform between keys so if you set a key at frame 1 to zero and at frame 10 you set the slider to 1 then at frame 5 the slider will be at 0.5 - which you may or may not want. If you want it to hold shape for a while before changing (*e.g. staying at zero until frame 7*), you'll need to set a key at the beginning and end of the time frame you want it to stay the same (*So you would set it a zero at the start, then zero again at frame 7 then to one at frame 10*). You can also edit the IPO curves (*use Shapes pop-up*) to really mess with the transformations.

4.45 Lip-Sync with Shape Keys

4.45.1 Lip-Sync with Shape Keys

Here I will attempt to explain my recent dealings with using Blender Shape Keys to produce convincing lip-sync (Lip-synchronisation, i.e.: "speech") for simple, humanoid characters.

This is aimed at people with an understanding of Blender fundamentals like *vertex loops*, *face loops*, *sequencer* and of course, Blender's new *Shape Key* system. If these terms mean nothing to you, then you may well struggle to keep up. If you're familiar with them then I hope this tutorial will prove to be a breeze and your characters will be speaking so fluently you'll have trouble shutting them up!

Other Lip-sync tutorials, if you can find them, recommend using other software like Magpie, Papagayo and others, but while I've no doubt they provide a valuable service and maybe make syncing easier, I will be using only Blender for this tutorial. I haven't tried using lip-

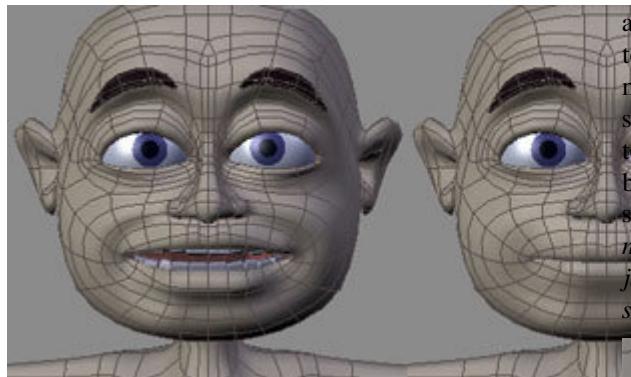
sync software yet so I can't really say if it's easier or not. You will find links to interesting audio files you can use for testing animation and lip-sync near the bottom of the page.

4.45.2 The hard work first

Setting up for Lip-Sync

First, set up your Blender screen so you have everything you need for this project. You'll need a front 3D Window, an Action Window, the Buttons Window (showing Editing - F9 - panels) and a Timeline Window. If you've got the room, a Side-view 3D Window would be handy too.

The basic sound units are called **phonemes** and the mouth shapes we use to represent these phonemes for lip-sync are called **visemes** (*or sometimes, phoneme shapes*) and there are many references for these around the web. Legendary animator Preston Blair created one of the **most popular viseme sets**, which is great for cartoon-style characters. Other visemes are aimed at more-realistic, humanoid characters. The shapes you choose depend on the style of your model. (*I'll try to provide some useful links later*)



Sample viseme shapes.

The first and most difficult step in good lip-sync is making the shape keys for these visemes. How well these are made ultimately determines the success of the animation and it is worth spending time getting them right, although they can be modified and tweaked later. So choose your favourite set of visemes (*or even look in a mirror and use your own face as a reference*) and start setting your keys. A model with good **topology** - especially well formed **edge loops** around the mouth area - will be a big help here!

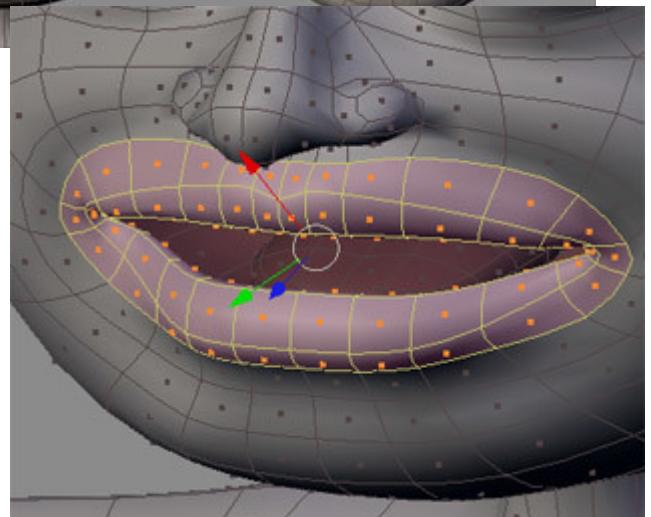
What on Earth are topology and edge loops?!?!

Topology refers to the way your 3D mesh defines the object, in this case your character's head. Edge loops refers to the flow of consecutive edges around the character's major features. Typically, good edge loops flow around the eyes and around the mouth in a somewhat deformed,

"circular" manner. Selecting and deforming a single loop of edges and vertices surrounding a facial feature is much quicker than having to individually select a lot of edges that don't flow naturally around that feature. Similarly, deforming a single loop or collection of related, nested loops is much easier and quicker than trying to deform a seemingly random set of edges. You can easily see these loops in the character screenshots above. The series of edges surrounding the mouth simply stretch from wide ellipses to almost circular to create a useful variety of mouth shapes. The faces defined by paired edge loops are referred to as **face loops**. The close-up image below shows selected face loops.

There are many tutorials on the web about these topics so if you need more information before proceeding then a quick search is probably a good idea.

Setting the basic viseme shape keys First, select your undeformed mesh and create your basis key. Go to **Editing** (F9) window and go to the "Shapes" panel. Press the "Add Shape Key" button. Enter and exit edit mode to set the basis key. Then create your first key **Key 1** and name it "M". Enter edit mode and if your character's mouth isn't already closed (*some people make them that way*) then close it by carefully selecting the faces and loops around the mouth. You will usually use Size-Z plus a bit of grabbing and shifting to achieve this. Don't forget to include the faces on the inside of the lips or the deformation will be unpleasant. When you're happy with the shape, exit Edit mode and there you have it. Your character can now say "Mmmmm" whenever you like. Test it by moving the Shape Key slider back and forth but make sure to leave it at zero before making more keys. (*If you made your character with a closed mouth then you can just add the new "M" key then enter and exit edit mode to set it.*)



Selecting face loops

For most new keys, you will select the basis key first then press "Add Shape Key" then make the required shape from scratch in edit mode. However, some mouth shapes

are very similar, like “OH and OOO” or “EE and SS” and it would be easier if you could start with something close to what you want rather than shifting everything from scratch every time. Luckily, Blender allows you to do just this. Once you've made your “EE” key, for example, you can select it and immediately press “**Add Shape Key**” then enter edit mode and the mouth will already be deformed and you only need to make subtle changes to it to make your “S” shape.

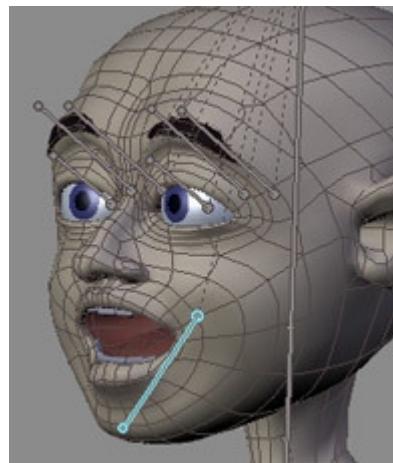
Remember that the keys you make are for sounds, not letters.

In general, you'll need shape keys for the following sounds

- **M, B, P**
- **EEE** (*long “E” sound as in “Sheep”*)
- **Err** (*As in “Earth”, “Fur”, Long-er” - also covers phonemes like “H” in “Hello”*)
- **Eye, Ay** (*As in “Fly” and “Stay”*)
- **i** (*Short “I” as in “it”, “Fit”*)
- **Oh** (*As in “Slow”*)
- **OOO, W** (*As in “Moo” and “Went”*)
- **Y, Ch, J** (“*You, Chew, Jalopy*”)
- **F, V**

These can often be used in combination with each other and with jaw bone rotations and tongue actions to also produce other sounds like “S”, “R”, “T”, “Th” and a short “O” as in “Hot” - or these can also be specifically made with their own shapes. This decision depends largely on your character. Start with the essentials and make others if you need them.

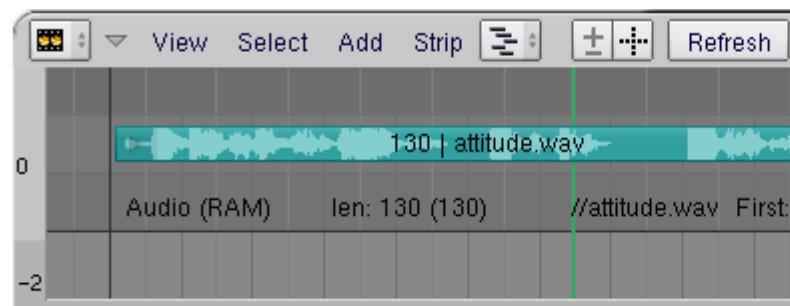
NOTE: I use one jawbone in my current character and this is also used to control the mouth shapes. It doesn't drive the shapes but it moves the bottom teeth and tongue (*which can also be controlled separately*) and the faces that make up the chin part of the character mesh. For some visemes, I move the jawbone into a logical position before adding the shape key. For example. I open the jaw for the “OH” key and close it for the “M” key. Later, when animating, the jawbone is animated along with the Shapes for a very convincing result.



Using jawbone with shapes

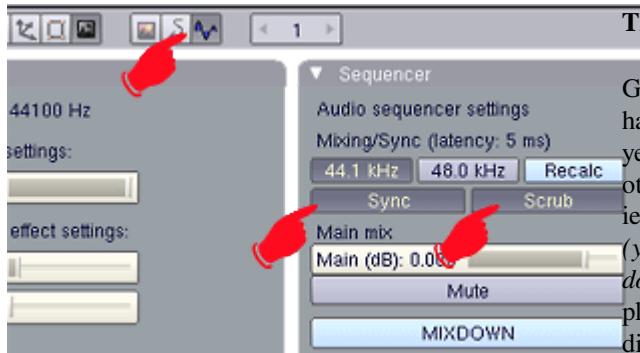
Load the audio Once all your viseme shapes are set the time comes to get your character to speak. (*Normally you would animate the body first and leave the lip-sync till near the end*).

If you haven't already done so, load your audio file into a Blender **Video Sequencer Window** and position it where you want it. Currently, Blender only supports 16 bit wav format audio files so you may need some editing or conversion software to process the file if it isn't in this format. “Audacity” is a good, open-source (free) editor that will suit this purpose and much more.



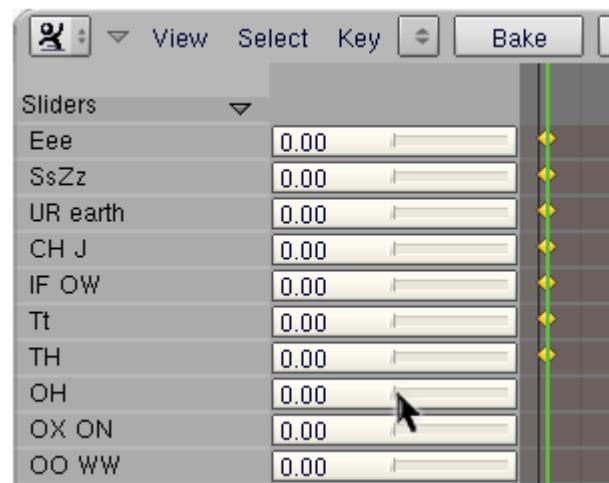
Blender Video Sequence Editor Window showing loaded Audio Strip

Go to the Scene Window (F10) and press the “Sound block buttons” button --(in the three buttons near the frame counter. It looks like an audio wave)--. Press the “**Sync**” and “**Scrub**” buttons. The “**Sync**” button makes the playback in the 3D window closely match the audio when you press **Alt-A** (*it does this by dropping image frames if necessary and it generally provides only a rough guide of how things match-up*). The “**Scrub**” button is important for lip-sync as it means that whenever you change frames, Blender plays the audio at that point (*Currently in some Blender builds you have to press Alt-A at least once to get this feature to start working*).



Blender Soundblock Panel

Select your character and your Shape Keys should be listed in the **Action Window**, in the order in which you made them (*I don't think they can be shuffled*). You will see a small triangle button labelled "Sliders" at the top of the list - press it to show the sliders for each shape. If you drag the mouse back and forth in this window, you should hear the audio play as you cross frames. This is how you will identify the main viseme/phoneme key frames.



Shape Key Sliders

4.45.3 Getting down and dirty

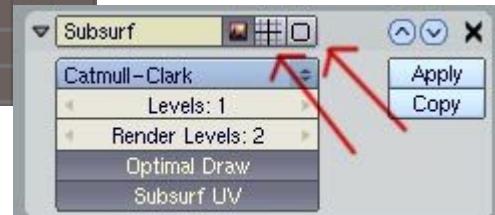
Before you begin your lip-sync, you obviously need to know what your character says. It might be worthwhile writing it down and even speaking it over and over, in time with the audio, to get a feeling for the sounds you'll be dealing with. Some sounds are what I'll call "**key sounds**" and others are almost dead, non-descript "in-between" sounds that fill in the gaps between the key sounds. Obvious key sounds are those where the lips close and those where they are tightly pursed or wide and round, other key sounds can differ with every piece of audio. Don't make assumptions about the shapes you'll use based on the words you know are there. What you are animating are the actual sounds - not letters or words (*Keith Lango has much to say about this on his website and I recommend reading it*)

The Timeline Window

Go to the **Timeline Window**. This window seems to have been largely overlooked in previous documentation yet it provides the basics for timed animation plus a few other goodies to make your animation life a little easier. Here you can turn auto-keying on and off at will (*you'd normally go to the hidden **User Preferences** window which is a pain and easy to forget*), navigate frames, play and pause the animation in the 3D window, turn Audio Sync on and off, set the start and end frames for the animation and **set frame markers**. This last feature is the key to our project.

In recent Blender releases, the audio "Scrub" feature works in most windows. As you scrub through the audio, listen for where the key sounds occur. As you hit a key sound, scrub back and forth to find where that sound commences. In the Timeline Window, press "M" to set a frame marker at this frame. The marker can be labelled by pressing CTRL-M in the Timeline Window while the marker is selected (yellow). Enter a sensible name for the marker that indicates what the marker is for (*like the phoneme sound and/or which word it starts or ends*). Markers are selectable and moveable and can be deleted just like other Blender items.

NOTE: Depending on the speed of your system, you may find you get more pleasing audio scrubbing and better 3D window playback if you turn off subsurf for your model and hide any unimportant parts of the scene on different layers. The fewer things Blender has to calculate as you scrub or play-back, the faster it can draw the frames to maintain sync with the audio.



Turn off

these buttons to disable subsurf setting in 3D window



Blender Timeline Window

Scrub through the whole audio - or sections of it in a long piece - marking and labelling all the key sounds.

Setting the keys

Now you have everything you need for your first lip-sync pass. Start at the first frame and click once on all your

Shape Key sliders in the Action Window to set them **all to zero**. Move through the frames from start to finish setting shape keys where you marked the key sounds in the Timeline Window. If your character has a jaw bone and tongue bone(s), you will need to set these where required as you go.

Trouble in paradise? A quick lesson in handling Shape Keys If you haven't set shape keys before you might notice one interesting dilemma - they have great memories! Once you set a slider to any value, it stays at that value until you set another value somewhere. The shapes change in a linear fashion between keys. At first, this appears to be a problem if you want to key "MOO" because after you set the "M" key slider to 1 (one), it will stay there, making it impossible to get your character to say "OO" while his lips are trying to stay shut. So, you have to set the "M" slider on the "M" sound, then as the audio goes into the "OO" sound, you must set the "M" slider back to zero and then set the "OO" slider to its full value.

This introduces another problem. After you set the "OO" key, your "M" sound is messed up because the mouth is now also being affected by the "OO" shape that follows it. So, you must make sure the "OO" sound is set to zero when you want the lips closed in the "M" position.

In general, you'll find yourself setting each shape 3 times

- once to determine where you want the mouth to begin moving to this shape (*slider set to zero*)
- once to set the slider at the desired level for this phoneme
- and once more to end this shape and move into the next one (*slider set to zero*)

The same principle applies to the jaw bone and tongue - 3 keys to each move.

Sliders	▼
Eee	0.00
SsZz1	0.00
UR earth	0.00
CH J	0.00
IF OW	0.00
Tt	0.00

Setting multiple shape keys

As you get more comfortable with this procedure, you'll find you can leave some shapes set longer or adjust them to different levels as they can provide some interesting mouth shapes when combined with other shapes.

Setting the in-between sounds

Once the key sounds are properly set, you should be able to scrub slowly through the Action Window and watch your character speak in time with the audio. It won't be perfect but it's a start. To fix his speech impediment, you now have to fill in the sounds between the key sounds. Mostly these will be dull vowel sounds ("err, uh, ah") and silence. These shapes are set in exactly the same way but here you'll have to really watch the 3D view as you set the sliders. Try combinations of sliders like "EE" and "OH" to get the perfect shape for each individual sound. You have to be guided by your character. Does he look like he's saying the sound you're hearing? (*Remember that exactly what's being said is not important - it's only the sound that matters*). Test each sound as you set it by scrubbing a few frames over and over and watching your character mouth the sounds.

All that's left, hopefully, is some polishing and tweaking. If it's not perfect then don't despair. Like other areas of animation, it can take a while to get a feeling for lip-sync and as the tools and workflow become more familiar you can pay more attention to the important work.

4.45.4 Putting it all together

When you're reasonably happy, it's time to combine the audio and video and watch the result. Since Blender can't do muxing (combined audio/video) you'll need to composite it with the editing software of your choice (OSX users can use recent versions of iMovie, virtualDub is often recommended for Windows users and Avidemux2 is often recommended for Linux users.).

What Blender can do is provide a **fully synced** version of the audio file the same length as the animation - even adding silence at the start and end if need be to maintain the synchronisation. To make this synced audio file, go back to the "Sound block buttons" panel and press "**MIXDOWN**". This saves a .wav file using the filename and location you entered in your output box (*you did didn't you?*) plus a frame count (*something like 'speech.avi0001_0400.wav'*). Then save your animation by pressing the "**ANIM**" button.

Combine the audio and video in your video editor and export a muxed file. You may find when you play it back that the mouth seems to be just slightly out of sync. This may be for two reasons: You're syncing per frame, the sound doesn't start exactly at a frame. The second reason is the way the brain processes faces and expressions and mixes it with sounds heard to comprehend speech. This comprehension phenomenon is barely understood and is a common challenge in animation, and some physiologists think our brains read lips and facial expression as a way of setting up to understand the context of sounds received and comprehending the meaning behind language. To solve it, you can go back into Blender, select the audio

in the Sequencer Window and move it one or two frames later, then press “MIXDOWN” again to create a new .wav file with a split second of silence at the start. Remix this with your video and watch the results.

From here on its all a matter of testing and tweaking until you're happy!

4.45.5 Using Blender to render Audio AND Video TOGETHER.

Ever since Blender 2.3, it has been possible to render video with the audio attached. Once you have your animation done, and are ready to render: In the buttons window, click scene (F10). In the format tab, choose FFMpeg. There will now be three tabs. Click on Audio. Click Multiplex, and choose your audio codec. Click Video. Choose your video codec (including AVI). Now, in another section of the window, there will be an Anim tab. Click Do Sequence. And click ANIM. When you watch the video, you'll notice that it has sound.

Note: The mac version of blender doesn't have the FFMpeg codec in blender by default.

4.45.6 Audio Files

You can find some interesting audio files selected for animators at animationmeat. These files come complete with a pre-marked phoneme sheet.

4.45.7 Note

One final note. Watch how the pros do this. When actors are doing voice overs for major releases their actions are recorded and even integrated into the final animation. If you have a digital camera, you may also try recording your own mouth performing your dialogue and approximating its positions to your animation. This can give you a great visual reference, possibly frames for frames if your frame rates match, and save you time.

4.46 Constraints

4.46.1 The Constraint

A constraint is what makes everything easier, magic, automatic, customised (add more words here) in a rig. It tells a bone or an object to do something special based on the position of another object, and the position of the constrained object itself. There are many constraint types for you to play with. Most will work everywhere but, the IK solver will only be available in the Armature Editmode or Posemode.

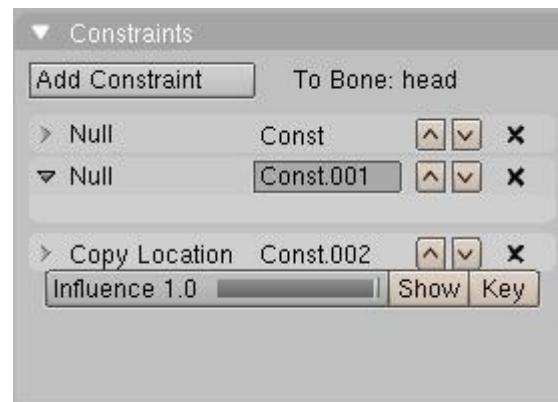
There are no strict rules to follow when using constraints. As long as they save you time and make everything work by itself. A constraint should never be “time-consuming” or difficult to use. Think about the animator who is going to work with this rig (it could be you!). So, do everything in a smart way.

It's possible to copy constraints from one object/bone to a bunch of objects/bones. A useful thing to know when doing a repetitive task like rigging all the fingers of a hand. Just select all bones/objects that you want to give a copy of the constraint, and then select the bone/object containing the constraints. Press **SHIFT + CTRL-C** in 3DView, and select Object Constraints from the popup menu. The idea behind this is to copy the constraints of the active object to the selection.

When working on an armature in Posemode, the bones will change color if they contain a constraint. Green for almost all, except for the IK constraint, which turns the bone Yellow.

4.46.2 The Constraint Panel

You can add a Constraint to an object or a bone by going in Object button window(F7) for objects and bones. Look for a Constraint panel like this (note, it's usually empty):



The panel also appears in Editbutton(F9) when you are in Armature Editmode or Posemode. So what you get:

- A button to add a new Constraint. The choice you have is listed down this page.
- When you add a new Constraint, A block gets added in the stack. The UI is almost the same as the Modifier Stack. Each block represents an entry. You can delete it with “X”, move it up or down in the stack, close or open it.
- Constraints are evaluated from first to last. So if you have two Constraints working on the same channel, let say Location, The last one will most probably win the chance to move the object. But...
- Most of the constraints have an influence slider to determine how much it will influence the stack. If

the last constraint has an influence of 0.5 it will mix the result with the one before.

4.47 Copy Location

4.47.1 Copy Location

- You can animate the influence of the Constraint by moving the time, changing the Influence slider and adding a key with the **Key** button.
- The **Show** button will display the influence IPO curve in an IPO window for editing. (The IPO window must be opened before pressing the 'show' button).
- You can change the name of the Constraint by clicking on the name when the constraint is open.
- By Clicking on the white jacket of the Constraint you select which one is active for edition, same as "show" button.
- If most of the Constraint you can enter the name of the Object you want to work with as a target or reference. For a bone, you need to enter in which Armature object it is, then an other field for the bone name will appear. When filling those fields, remember you can use autocompletion using **TAB**.

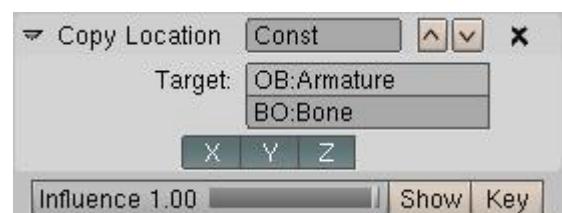


The Copy Location constraint does as the name states: it will copy the location of the target to the source (constrained object). Less Influence will drag the constrained object less and less to the target.

If it's an armature, a new field will appear to let you tell which bone will be the target. Don't forget TABKEY completion while writing the name of your object/bone!

You can tell Blender to work only on the selected axis. Many uses are possible :)

4.47.2 The Constraint Panel



- The Target field will let you select which Object the constraint holder will follow.

4.46.3 The Constraint Index

- Copy Location
- Copy Rotation
- Track-To
- Floor
- Locked Track
- Follow Path
- Stretch-To
- IK Solver
- Action

4.47.3 Where To Use It

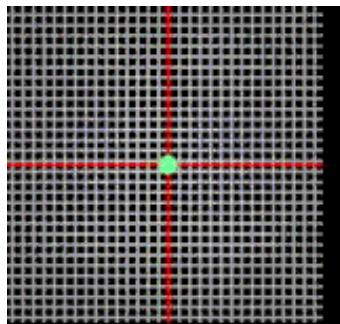
Most of the time this little constraint is useful to stick objects to one another. By playing with the Influence you can tell when it will work, when it will remain motionless.

A good use of it is to ask a character to pick up something. By having a bone or empty for each side of the relationship (hand <-> glass), as the hand approaches the glass, you can align the two empties and fire the constraint up (1.00) to stick them together. You add another child-bone in the middle of the hand to tell where the glass will be. Thus moving the hand will move the glass. On the side of the glass just add an empty and make it parent of the glass. Add a copy location to the empty pointing to the bone in the hand we just did. There you go. Of course

when the hand rotates the glass will not. For that you will need to add a Copy Rotation Constraint.

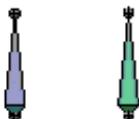
Before Blender 2.40, the above method was a good way of faking parent relationship without rotation. But now we have the hinge option which does the same.

Create this kind of tracking device using the X Y Z toggle button



4.48 Copy Rotation

4.48.1 Copy Rotation



This constraint copies the rotation of the target. As simple as that. It can be an object or a bone. As you can see in the example, only the rotation gets copied.

4.48.2 The Constraint Panel



- You have 3 buttons to select which axis get copied over.

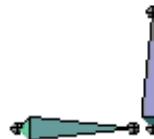
4.48.3 Where To Use It

Can be used with Copy Location to fake parent relationship. As you can key the influence you can make a character pick something up and hold it in his hands.

You can also use this to align a plane with a 2D effect on it to the camera at all times. This works better than pointing it at the camera in some cases, such as a ring of atmospheric halo around a planet, where you don't want it disappear behind the planet.

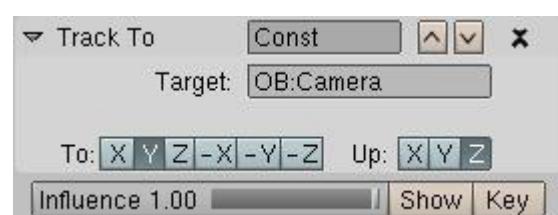
4.49 Track-To

4.49.1 Track-To



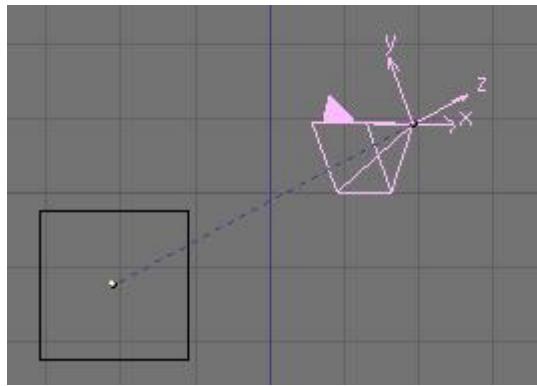
The Track-To constraint lets you influence the Rotation of the constrained object by making it track a target with one of the constrained object's axes.

4.49.2 The Constraint Panel



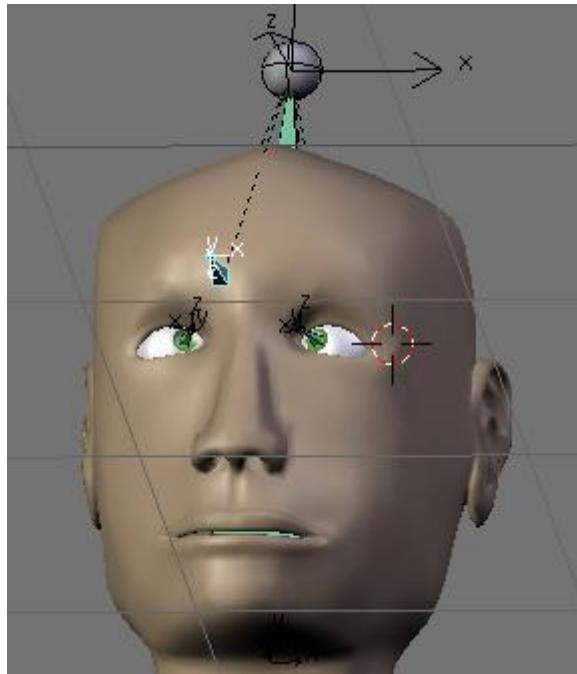
- You can enter the name of the target you want to track.
- You can select which axis is going to track the target.
- You can select which axis is going to stay up.

4.49.3 Where To Use It



A good example of use is the make a camera track an object. The setting to use on a camera is track: -Z and up: Y. You can turn Axis drawing in objectbutton window to help you choose the good axis.

Another example with armature would be the eyes of a character:



4.50 Floor

Using the floor constraint will keep a bone from passing through an object from a given direction. It is useful when making floors of course but also when making walls and other items which are obstacles for the armature.

There is also an offset value which is very useful when say for example you have a foot where the mesh stretches down below the actual tip of the armature you can then use the offset to make the bone stop before it actually reaches the obstacle object.

4.51 Locked Track

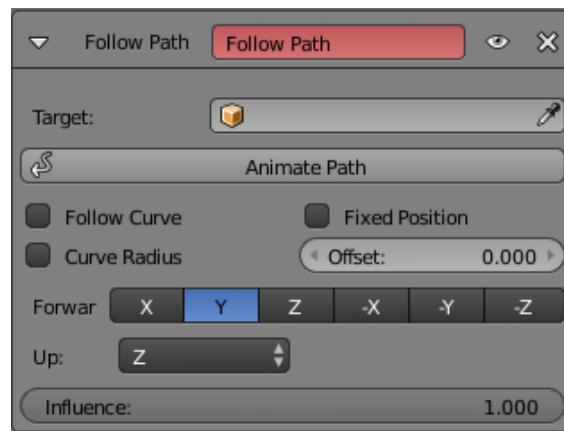
4.51.1 Locked Track

This page is just like Track To, only you can lock an axis to point upwards.

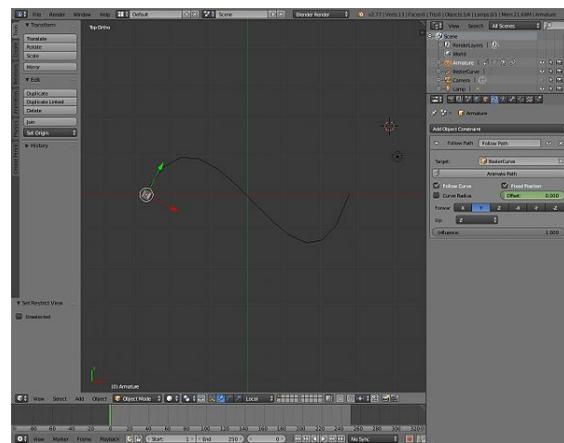
4.52 Follow Path

4.52.1 Follow path

Follow path is a constraint which creates an animation in which the Object follows a Curve. You of course have to constraint it to a curve of some kind.



4.52.2 The Constraint Panel



When you added a curve you've to click the "Animate Path" button, otherwise nothing will happen.

The "Follow Curve" checkbox is there to make the objects Forward Axis, here "Y", point in the direction of movement. Of course here you'll find the Forward axis choices and Up choice, weird they still got different layouts, but don't bother. Finally, the "Influence" slider.

4.52.3 Where To Use It

It can be very handy when you animate a roller-coaster or something that has to follow something else perfectly.

Here's an animation of a follow path

4.53 Stretch-To

4.53.1 Stretch-To

Stretch To causes the affected object to scale the Y axis towards a target object. It also has volumetric features, so the affected object can squash down as the target moves closer, or thin out as the target moves farther away. Or you can choose not to make use of this volumetric squash-'n'-stretch feature, by pressing the NONE button. This constraint assumes that the Y axis will be the axis that does the stretching, and doesn't give you the option of using a different one because it would require too many buttons to do so.

This constraint affects object orientation in the same way that Track To does, except this constraint bases the orientation of its poles on the original orientation of the bone! See the page on Tracking for more info. Locked Track also works with this constraint. Options

R

Pressing the R button calculates the rest length as the distance from the centers of the constrained object and its target

Rest Length

Rest Length determines the size of the object in the rest position

Volume Variation

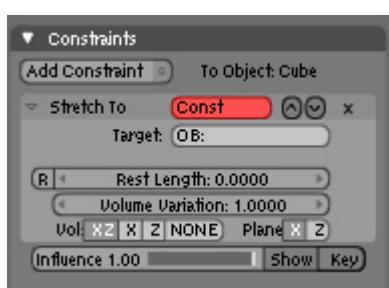
Volume Variation controls the magnitude of the effect

Vol

The Vol: buttons control along what axes the volume is preserved (if at all)

Plane

The Plane buttons define which local orientation should be maintained while tracking the target



4.54 IK Solver

4.54.1 The IK solver



The IK solver constraint is a wonderful tool for all animators. IK stands for "Inverse Kinematic" and is the opposite of FK (Forward Kinematic).

- FK: You have a dependency to the root of the chain. In Blender, a FK chain is a chain of bones connected together. You have to animate the bones rotation one by one to get it animated. It takes longer, but gives you entire control over the rig.
- IK: Both ends are roots. All bones in the chain are trying to rotate to keep both ends on targets. Now this Constraint got most of the attention during Animation refactoring, hopefully we have a lot of toys to play with now.

The IK solver has a special shortcut **in Posemode** to be added easily to a bone. If you select a bone and press '**CTRL-IKEY**', You get a little menu asking for more info on the new constraint, the target: to a new empty object or without target. It's now possible to work without target. Though you have less freedom (no rot feature, difficult parent relationship).

You can also select the target and then the IK constraint holder and press **CTRL-IKEY**. With this way of selecting ensure that your target is selected, but the bone you want to apply the constraint to is active (the last one selected). The menu will then let you add a constraint to the current bone with a target. If the target would itself be part of the IK chain, you get an error message - so make sure the target bone is not connected to the bone you want to add the constraint to.

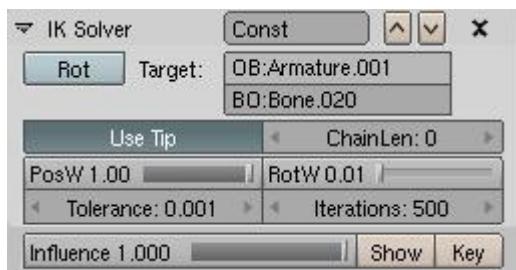
It's also possible to remove all IK constraints from selected objects or bones with '**ALT-IKEY**'.

Q: '**CTRL-IKEY**' doesn't seem to do anything

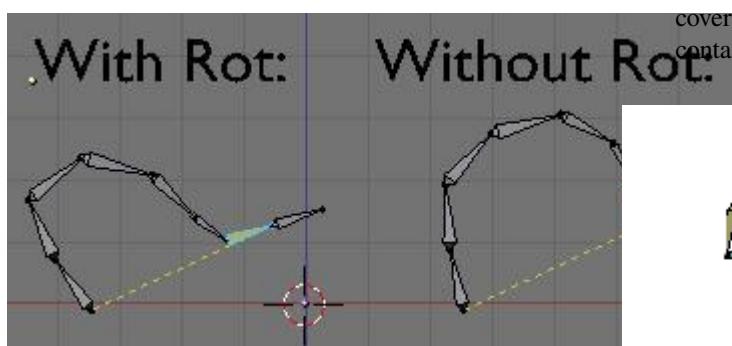
A: Either the 3D Window is out of focus (R-click in empty space to solve) or you're not in Pose Mode ('**CTRL-TAB**', selected bone will be magenta in color)

A: In the 2.48 version of Blender, the shortcut is '**SHIFT**-**I**'

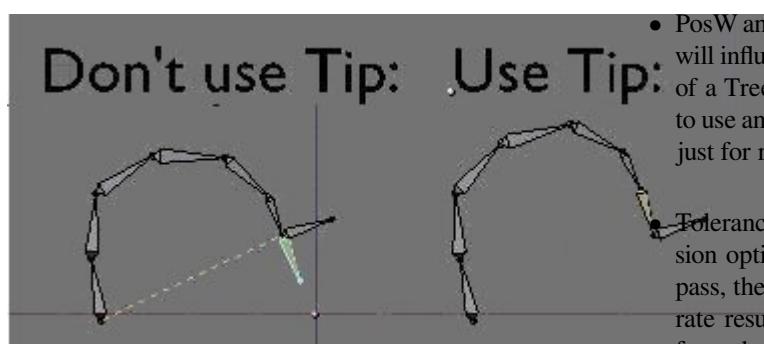
4.54.2 The Constraint Panel



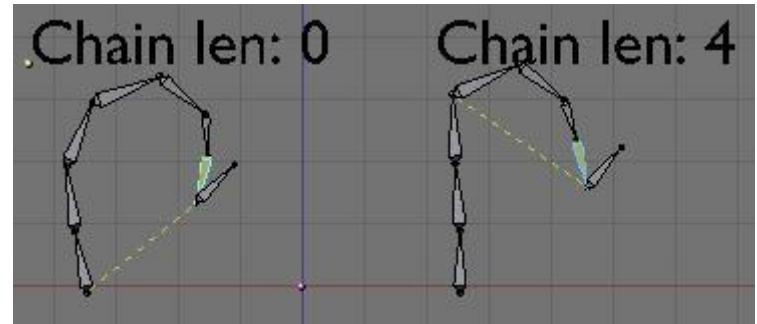
- You can rename the constraint.
- You can select which Object or bone will be the target. Don't forget Tab completion.
- The Rot button let you tell Blender to use the rotation of the target to influence the rest of the chain:



- The Tip button lets you tell Blender which part of the bone is the target, the Tip or the Root. It's interesting to use tip, because this way the Bone holding the IK constraint can be used to deform geometry.



- Len lets you tell Blender the length of the chain the IK solver will try to rotate. If set to 0, the entire chain will enter in the constraint. If for example the len is 4, only the 4 last bones of the chain will try to touch the target.



- Also If you set len to 0 and your chain's root is a child of another bone, The IK solver will reach and rotate all the bones until it gets to the end of the parent relationship. If all the bones are linked up to a master root, then all other sub-branches will be affected. If there is another IK target in other sub-branches of the rig, Blender will try to mix them. This concept of multiple IK targets in a rig is called Tree IK and can be used to get completely automated animations. For example like a doll: if you pull one hand, all the body will follow. In the 3D-view you'll see a yellow line from the IK solver to the root of the chain it covers. This line appears when you select the bone containing the IK solver.



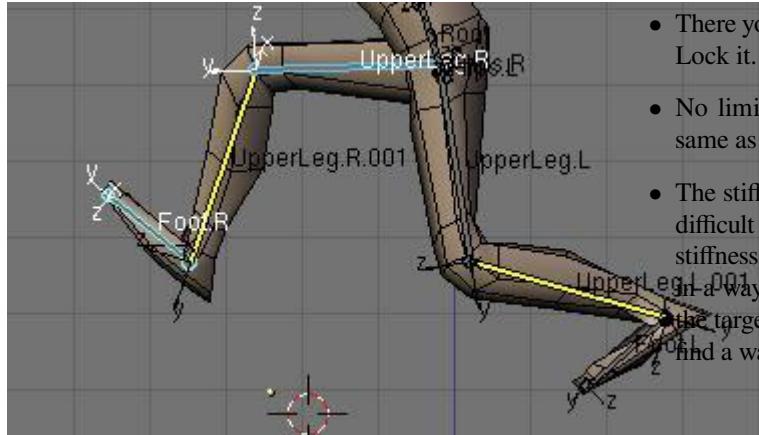
- PosW and RotW let you tell Blender if this IK solver will influence the final result more or less in the case of a Tree IK setup. With these options it's possible to use an IK solver just for location and an other one just for rotation.

Tolerance and iterations are performance and precision options. IK solving is done in more than one pass, the more passes you calculate, the more accurate results you get. The tolerance is the distance from the IK solver to the target you can accept. If Blender manages to place the target near enough, it will stop doing iterations. The Iterations value is a hard limit you set to limit the time blender can reach on each IK solver per frame. Try to set it to a very low value to know why Blender needs more than one pass ;).

- You can set the general influence this constraint will have over bones, and it's animatable.

Lock X Rot	Lock Y Rot	Lock Z Rot
Stiff X: 0.000		Stiff Z: 0.000
Limit X		Limit Z
Min X: -94.7		Min Z: -64.8
Max X: 118.4		Max Z: 88.6

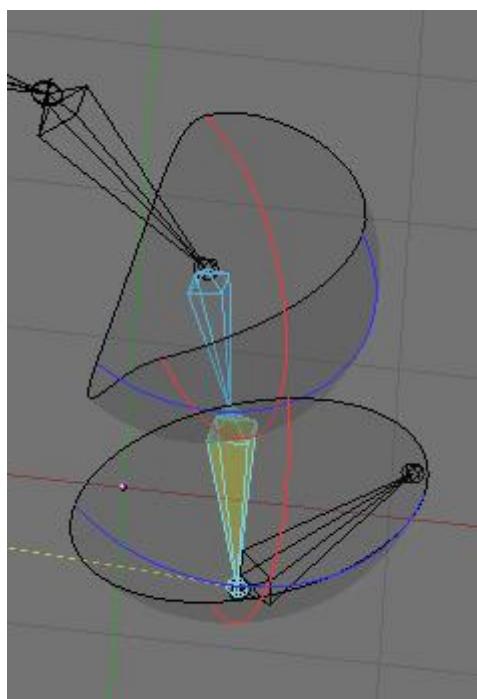
4.54.3 Where To Use It



- There you can set a limit on each axis, or completely Lock it.
- No limit gives it complete freedom (which is the same as [min:0 max:360] or [min:0 max:0]).
- The stiffness lets you tell Blender if an axis is more difficult to rotate than the rest. If all bones have a stiffness of 1 on X and you try to curve that chain in a way that all bones need to turn on X to follow the target, the Solving will find really weird poses to find a way to touch the target without rotating on X.

In any chain of bones you don't want to animate by hand but you want both ends to be at precise location. The best example is a leg: The leg is connected to the body and to the foot. You don't need to animate the 2 bones in the legs, just place the body and the foot, the leg will follow automatically.

4.54.4 Degree Of Freedom



In DOF it is now possible to set it for bones in an IK chain. This way you can set what will block where. This is very useful when doing a mechanical rig as you can limit the move or better, lock completely an axis.

4.55 Timeline Window

4.55.1 Timeline Window

The Timeline window is below the 3D view in every standard project. You can see where keyframes are for the selected object and scroll through the timeline

4.56 IPO Window

4.56.1 Graph Editor

The Graph Editor is meant to work with curves, which mostly represent key-frame transitions or drivers.

4.57 Data Type

4.57.1 Dope Sheet

The Dope Sheet is a window in which you can move, very easy, keyframes from one frame to the other. They are visualised with clear diamonds.

4.58 NLA Window

4.59 The NLA Window

It's quite easy and maybe that's why there's no specific tutorial. Let's say you want to make two actions, AC:Hit and AC:Kick. Start with posing Hit and an Action will automatically be created in the Action Editor consisting of all the Bones that use Action IPO's. That's done so

return to Frame 1 which will be your default Stance of AC:Hit.

Now, in the Action Editor, click the X (delete) next to AC:Hit and the datablock menu will disappear. (If you Add New instead of deleting then it will copy selected bones to the new action and you don't always want that). If you want a new default pose for AC:Kick, then Pose it or the same stance will be used as was the default in AC:Hit. Pose and Keyframe your Kick action and name it.

Over in the NLA Editor you can now use Shift-A to add NLA-Strips of your Actions, Grab and Scale them and use the Transform Properties tab to input how they Blend.

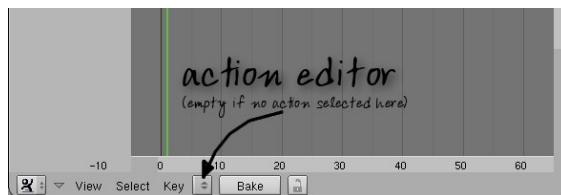
If you select an Action with the dropdown menu, at first its name will appear in the NLA window along with its keys. To make this Action into an NLA strip, point at the Action's name in the NLA and press CKEY.

Close any open Actions by clicking the scary X in the Action Editor. If you don't do this, only this action will play. Now in the NLA editor, play the animation.

If you see any keys (diamonds) in the NLA window, instead of strips (rectangles), you're still editing an action. It's so much easier if you have both the Action and NLA windows open so you can see whether an Action is open or not.

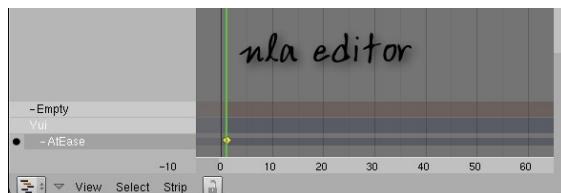
4.59.1 Walkthrough

With no Actions selected, both the Action Editor and the NLA Editor appear empty. Here, the NLA Editor window does list one Object called **-Empty** because that object is not an armature but it has some IPO curves attached.



empty action window

Select an Action you've already made. Here, an Armature named **Yui** has bones involved in a one-frame action called **-AtEase**.

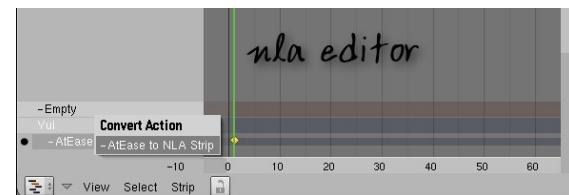


nla window with action

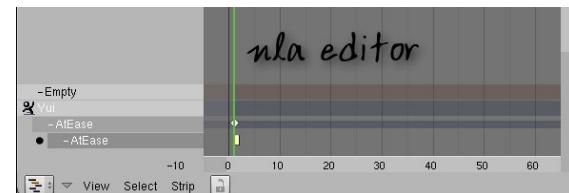


action window with action

Convert the listed Action to an NLA Strip in the NLA Editor by pressing the CKEY with the mouse hovering over the Action to be converted. No change in the Action Editor; it is still available as an Action.



nla window converting to a strip



nla window after conversion

Once converted, note the changes in the NLA Editor. The Action icon appears next to the Armature's name: **Yui**. This is actually a button though it does not look like it, and you can toggle it between the Action and NLA Strips icon by clicking on it: **Yui**.

4.60 Introduction To NLA Editor

4.60.1 NLA (Non-Linear Animation)

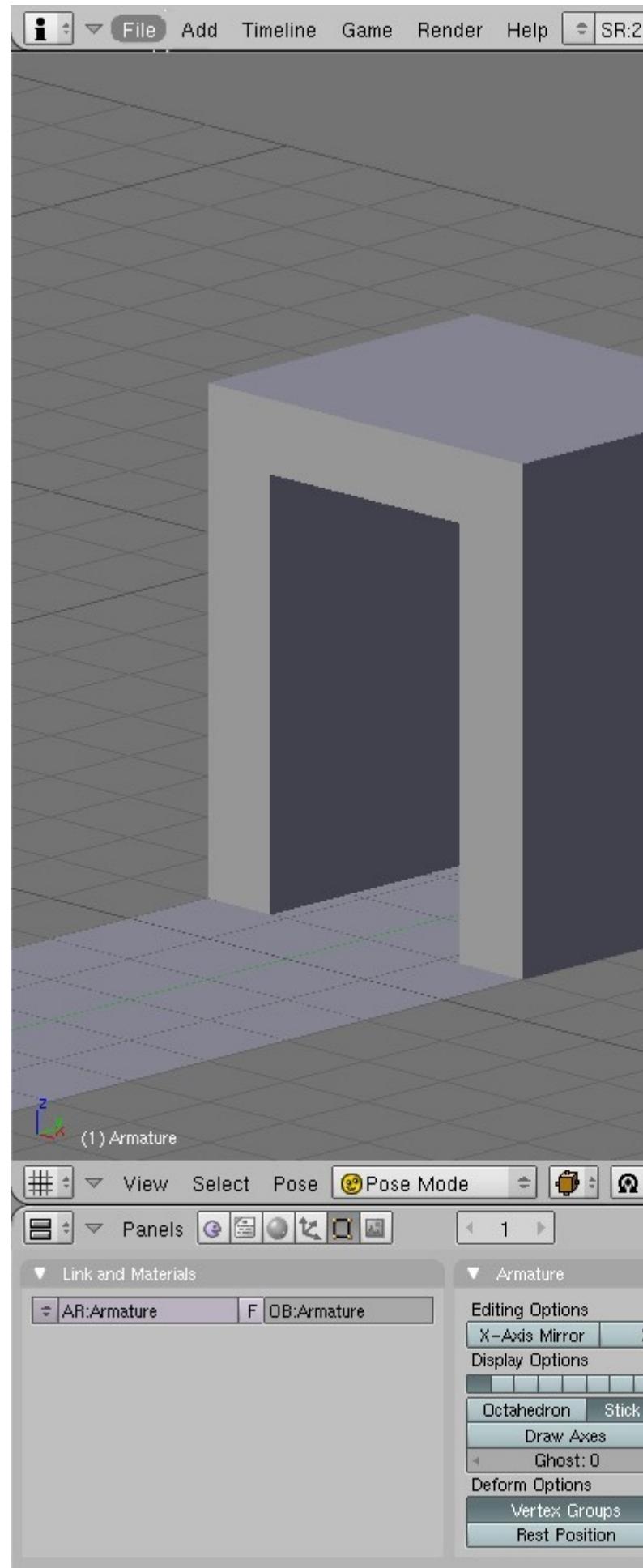
WARNING-This page assumes the reader to understand the IPO window and the Action Editor window, as well as rigging a character with an armature object. This tutorial will make little sense without this previous knowledge.

Imagine this--yourself, sitting at your computer late at night, beating away at a huge blender animation. You didn't think it would take as long as it did, but you lost your composure staring into the jungle of colorful IPO lines, little white and yellow Action diamonds, and that annoying green current frame line. You know that the NLA window would help you make sense of it all, but you are afraid of opening that Pandora's Box because of

the problems that will follow it. Never fear, for this tutorial will clean up all those problems and revolutionize the way you blend for the rest of your life.

4.60.2 Setting up the Scene

It will be easier for you if you start with a small demo file than if you go straight for the big prize. Give a character or other armature-rigged object a few SEPARATE actions (remember to name them something distinct, like "Walk" or "Run" as you always should with *everything*). For your own sanity, you will want to have a path or an IPO that correlates with the actions in question. All of the blender screenshots I will be using come from a file with a very basic stick person, rigged with an armature skeleton. He has two actions; a normal walk cycle, and then another, hunched over one, as if to pass beneath a low ceiling. If you use this idea in your own test file (and I thoroughly recommend it) do yourself a favor and give them both the same stride length! Give this guy an IPO or a path to follow that keeps his feet from slipping, and make sure that it has a linear interpolation mode. The rest of the scene consists of a floor-plane and an arch too small for him to walk under (hence the crouching walk). Place this arch exactly one walkcycle away for now, we will move it around later.

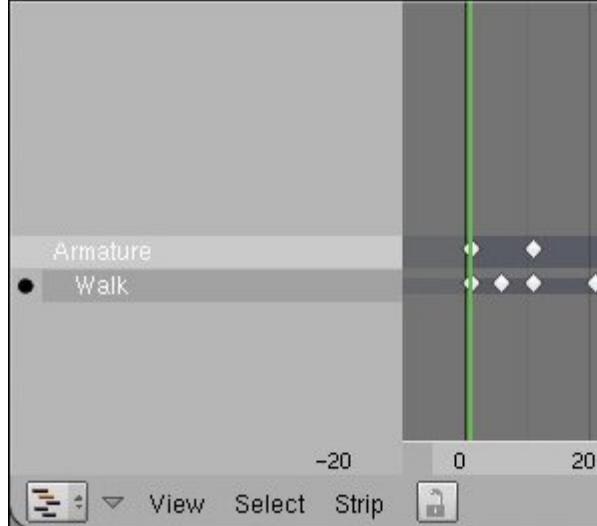


Complete the scene before going on to the next step.

4.60.3 Adding Action Strips

Now, he moves forwards as you scroll through the frames, but how do we get him to walk forward, duck, and walk under the arch, too? First of all, you need to select the armature object and create a link to the normal walk. By pressing the up or right arrows, or pressing “alt+a”, you should be able to see him walk up to the arch, stop walking, but keep sliding through, with his head sticking out the top. As ridiculous as this seems, right now, however, you are on the right track. Split your window now and open the NLA window with .

At first glance, this window looks almost identical to the action window we all know and love. This is not totally off of the mark. The NLA window is, in essence, an abbreviated version of the action and IPO windows.

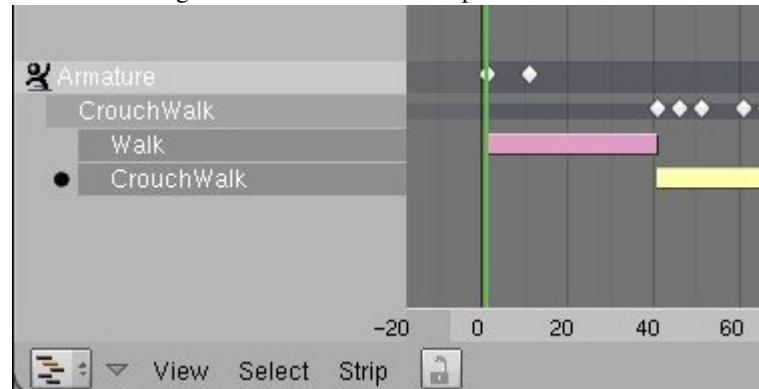


This is what you should see in the NLA window.

You see a space with the name of your armature (named “Armature” in my demo) and in that row you see a few key diamonds. These show the IPO you have put on the person. I made it so the IPO extended on forever, using the first ten frames as a guide. In a subset beneath the armature object, you see the name of an action. The bullet next to it means that the armature is currently linked to that object. In that row, you see key diamonds, and those correlate with the keys you put into the action in question. If you switch actions in the action window, you see that the NLA window also changes. But how do you get it to keep the first and play the second later?

Go to the first frame you want him to walk in (most likely number one) and place your cursor in the NLA window. Press “shift+a” to see a dropdown menu. Pick the name of the first walk cycle. You should see a new subset space appear, with the name of the first walk in it. This space should be occupied by an action strip. Two colors of action strips are yellow and pink, yellow

meaning selected and pink meaning deselected, just like IPO vertices. This strip takes up the amount of frames the walk did. The NLA window has now saved your first action. Now to toggle to the next action, move the frame line to the end of the first strip, then press “shift+a” again, this time clicking the other action on the dropdown menu.



The result.

4.60.4 Getting the Strips to Play

Now, as cool as this seems so far, if you press “alt+a”, you will be disappointed and confused. The animation will only play one of the actions. This is a simple problem to fix. If you look carefully, you will see that the armature is still linked to the action it played. This data is overriding any other data from the NLA window. Press the “X” in the action window right next to the name of the action. Press “alt+a” again. It works!

This is, however, a jerky and instantaneous switch from one action to the other. Making it less weird is easy. Press the “N” key with your cursor in the NLA window. You will see a small box with a bunch of functions in it (this will be explained in detail on later pages).

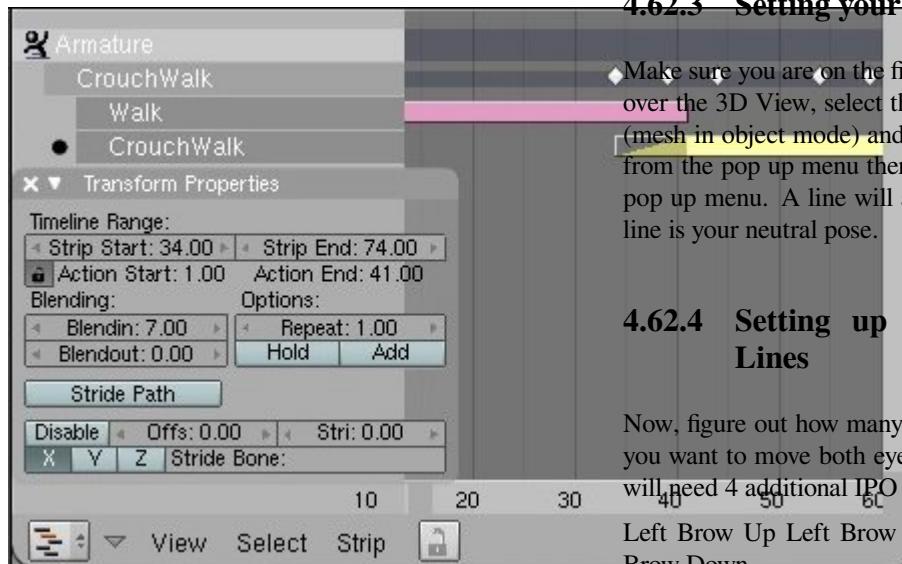


The Properties window.

This little box is the lifeblood of the NLA in blender. It contains all the tools you will need to be able to run this feature smoothly. I will illustrate a few of them right now, but most others will be shown in the section on this window.

Make sure you have highlighted the **second** action strip, and look at the space where it says “Blandin”. The number there is the number of frames blender will use to

smooth the transitions *in to* the highlighted action. Depending on how fast your character is walking, you may need to change the number a bit, but I set my “blendin” for seven. Whatever number you used, however, move the second strip *back* that same amount of frames. The wedge on the end of the action strip should end as the other strip ends.



The NLA solution.

4.60.5 Conclusion

You now have the necessary skills to complete much more involved and complex animations. You can, however, increase this even more by continuing to read the other NLA tutorials in this wikibook.

Good Luck!

4.61 The Stride feature

At the moment, one of the best documentation for Blender Stride features can be found here: [Blender Stride Tutorial](#), which takes on where the [Official Blender Stride Page](#) takes off. Good luck! A simpler and shorter, but still very good tutorial can be found here: [Blender wiki: NLA Editor and Stride Path](#).

4.62 Relative Vertex Keys

- Letters in brackets i.e.: (z) mean there is addition information at the bottom of the page.

4.62.1 Introduction:

This tutorial is meant to stop all the RVK (Relative Vertex Keys) questions.

4.62.2 Window Layout:

Set the left half of the screen as 3D View. The other half is divided in two. The top is Action and the bottom is IPO (set to vertex display).

4.62.3 Setting your Neutral Pose

Make sure you are on the first frame (a). With the cursor over the 3D View, select the mesh you want to animate. (mesh in object mode) and press the I key. Select Mesh from the pop up menu then Relative Keys from the next pop up menu. A line will appear in the IPO view. This line is your neutral pose.

4.62.4 Setting up your additional Pose Lines

Now, figure out how many key frames you will need. If you want to move both eyebrows up and down then you will need 4 additional IPO lines.

Left Brow Up Left Brow Down Right Brow Up Right Brow Down

Press the up arrow (cursor key) to move to forward 10 frames. Press the I key while over the 3D View and select Mesh. Repeat until you see a total of 5 lines in the IPO window.

4.62.5 Set your Poses

Right click on the Neutral pose line in the IPO window. This sets the mesh to the neutral pose. Now Right click on the next line up in the IPO window. Enter edit mode in the 3D View and move the vertices as desired (in this case you will be moving verts to get the left Brow up pose). Press Tab to exit edit mode. Now right click your Neutral pose line in the IPO window. You will see your object in its neutral state. Right click the next line up and you should see the changes you just made to your object. Set up all your mesh poses following the above instructions.

4.62.6 Name your Poses

Right click on the Key names in the Action window. Change the name and click OK.

4.62.7 Time to Animate (b)

Click on the arrow next to the Sliders text. This will give you access to the pose sliders. Move to frame 20 to start your action. Move the pose slider but release the mouse when set to 0. Now move 10 frames forward and move the same slider to 1.00 (maximum). Use this method to

set up all your actions(c). Remember to add a 0 value frame to end the pose.(d).

4.62.8 Adjust your Slow in & Out

In the IPO View select from the menu to find the IPO curves. You can get back to the Pose lines by selecting KeyIPO from the same menu. Right click the spline you want to edit and press TAB to enter edit mode. Move the handles to adjust slow in/out.(e)

(a) In this case moving to a frame has nothing to do with animation. It is done so that your pose lines are separate from each other. (b) Select your key frame marker and use the usual commands to move $<\text{g}>$ and duplicate $<\text{d}>$ them. (c) Be subtle by not pushing the slider all the way to 1.00. (d) Try overlapping your poses. (e) When setting slider values they can sometimes go into the negative value. This will give you weird results. Although sometimes they can make your animation more interesting. To fix this edit the IPO, select the point where the line dips below zero and press the V key. Do the same at the other end of the curve if needed.

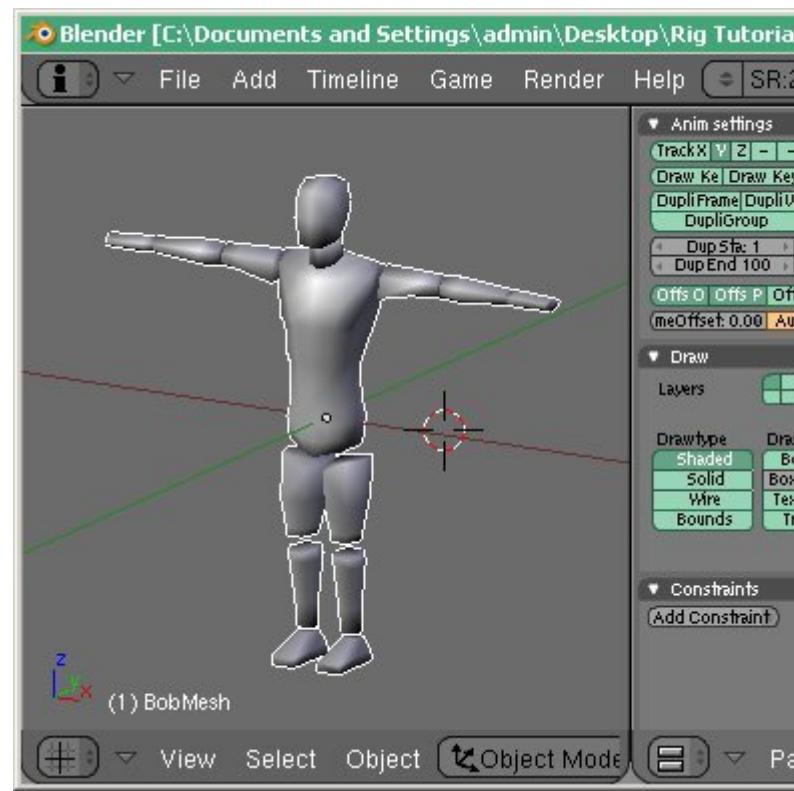
Warning! Blender has a limit to the number of verts you can use.

[Click here](#) to read the advanced animation tutorial guided tour.

4.63 Working Example: Bob

In this tutorial, we are going to be constructing a fully functional, humanoid character, with a complete rig, and we are going to animate him performing a walk cycle.

4.63.1 Getting Started



- Build the Rig
- Deform the Mesh
- Create a Walk Cycle

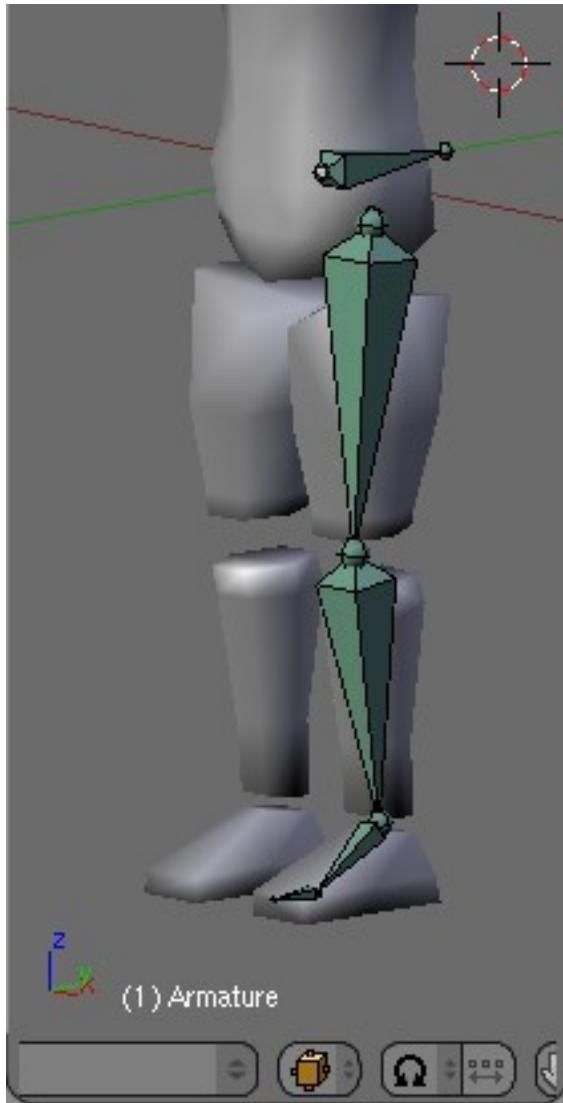
4.64 Building the Rig

If you think you already know what a **rig** is, then you probably need to read the definition of “rig” and “rigging” before we get started. It’s important to note that an armature is not a rig, but a rig can be an armature. Assigning a mesh to be deformed by an armature is not rigging.

A **rig** should always be designed for the types of animations your character is going to be performing. Only make your rig as complex as it needs to be to allow for the types of actions you need.

To make everything with the armature easy to deal with, we’re going to make our character in the crucifix pose. If he’s not, you will have headaches trying to deal with bone roll angles. Once Blender can easily allow the user to roll the bone to align with a roll target, then I’ll edit this tutorial for that. But in the meantime, we will use vertical legs and horizontal arms. We will build the legs from the side view and arms from the top view.

Center the cursor (shift+c) and add an armature. In **Object Mode**, press alt+r to clear the rotation. You have to have a bone for the hip, and it needs to stick out of his



In front view, center your cursor and select pivot point

4.65 Deform the Mesh

This is where you do the animating.

4.66 Create a Walk Cycle

This is where you create the walk cycle.

4.67 Working example: Piston, Rod and Crank

4.67.1 Piston, Rod and Flywheel

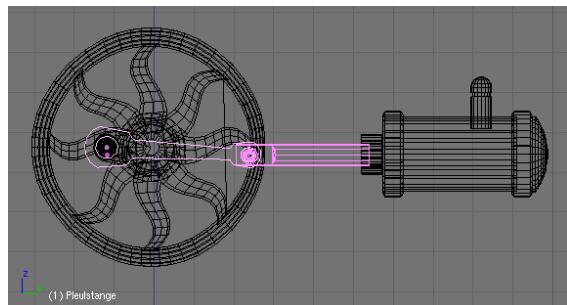


Image 1: Surprisingly difficult: the animation of Piston [Kolben], Rod [Pleuelstange] and Flywheel [Schwungrad]

front or his back, so take your pick, because they both look bad. Don't point the bone upward at some odd angle, we need to be able to roll the hips easily, and to that end, we will place the bone horizontally.

In front view, place the cursor and add a bone. **IMAGE**

In side view, move points and extrude them until your chain looks like this. Note the slight bend in the knee. This is very important! **IMAGE**

Snap your cursor to the root of this chain (shift+s) and add a bone. Now select the points at the hip joint and the ankle joint, and snap the cursor to the selection. **IMAGE**

Select the tip of the newest bone and snap it to the cursor. **IMAGE**

Now give these bones some names. It's a good idea to use the same names I do to avoid confusion, since I will refer to the bones by name. Select upperleg.l and then shift+select leg.l, and press ctrl+p to make upperleg.l the child of leg.l. Do this again, but make leg.l the child of hip. **IMAGE**

A setup like that shown in **Image 1** is e.g. typical for a steam engine. It is surprisingly difficult to animate this setup, because one has to translate a rotating movement (of the Flywheel) to a horizontal movement (of the Piston). This alone wouldn't be too difficult, but the Rod has to stay attached to the Piston. The shown setup uses three empties and two armatures, each with two bones.

The horizontal direction is the Y-direction, if this is different in your model you have to exchange Y with your horizontal direction.

Setup

- Use **Ctrl-A->Apply scale and rotation** on all mechanical parts in object mode.

This will spare you from bad surprises if you rotate the flywheel afterwards.

We need an object which follows the rotation of the flywheel. We use an Empty, which is parented to a Vertex.

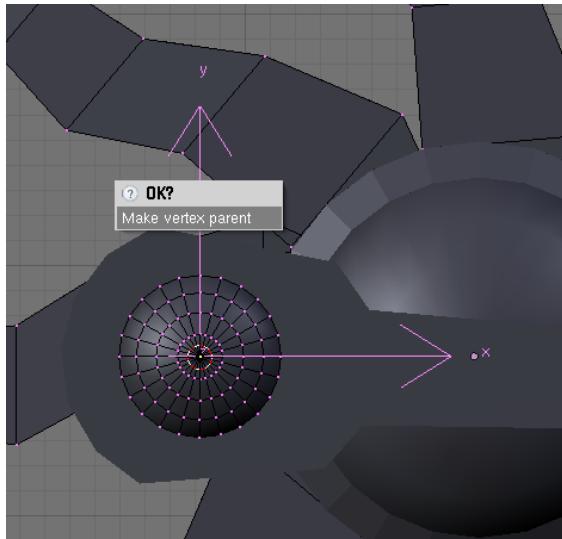


Image 2: Make vertex parent of an Empty.

- Change to edit mode of the flywheel. Select a Vertex in the middle of the crank.
- Move the cursor to the Vertex (**Shift-S->Cursor->Selection**).
- Change to object mode and insert an Empty (*ERotation*).
- Change back to edit mode of the flywheel. The vertex is probably already selected, now select with **Ctrl-RMB** the Empty and press **Ctrl-P->Make vertex parent** (**Image 2**).

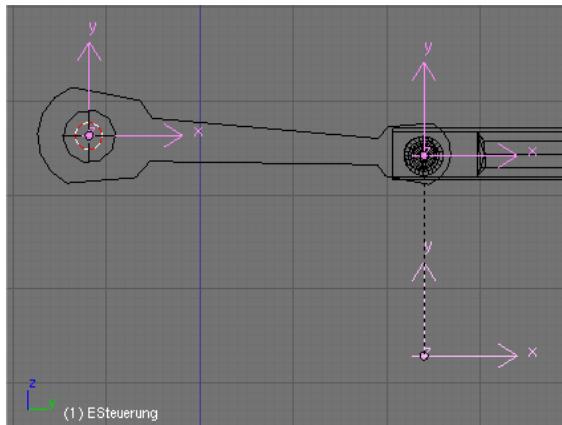


Image 3: The positions of the three Emptys.

We need two more Emptys:

- Place the second Empty (*EGelenk*) in the center of the joint between Rod and Piston.
- Place the third Empty (*ESteuerung*) near to the joint.
- Now select at first *EGelenk*, than **Shift-RMB** *ESteuerung* and parent *EGelenk* to *ESteuerung* with **Ctrl-P**.

We're going to insert two armatures, each with two bones. The first armature, helps translate the rotation to a horizontal movement. The second armature drives the Piston and the Rod. The cursor should be positioned at the Empty *ERotation* (like in **Image 3**), else set it there.

- Add an armature (*AHelper*). The root of the first bone (*BHelper1*) has the same position as *ERotation*, the tip has to be positioned to the Empty *EGelenk* (select Empty first, *Cursor->Selection*, than select bonetip, *Selection->Cursor*).
- Extrude another small bone (*BHelper2*) in horizontal direction.

BHelper2 is automatically *Child* of *BHelper1*. The second armature that is driving the movement of piston and rod will be inserted in the opposite direction, i.e. from right to left in our example.

- Position the cursor on the piston.
- Add an armature (*AGestaenge*).
- Move the tip of the first bone (*BKolben*) to *EGelenk*.
- Extrude the first bone and place the tip of the second bone (*BPleuelstange*) to *ERotation*.

Connecting the two bones will ensure that the rod cannot disconnect from the piston.

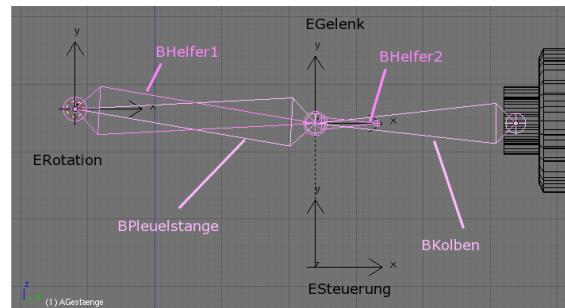


Image 4: The alignment of the two armatures.

Now we're going to add the “logic”.

- Add a *Copy Location* constraint to *BHelper1* which points to *ERotation*. To do this change to the pose mode of the armature *AHelper* and select *BHelper1*. In the *Constraints* panel click *Add Constraint->Copy Location* and type *ERotation* into the *Ob:* field.
- *BHelper1* needs a second constraint: *Track To* to *EGelenk* (**Image 5**).

Now the root of the bone *BHelper1* follows the rotation of the Flywheel. Its tip follows the joint, but with slight deviations upwards and downwards. The position

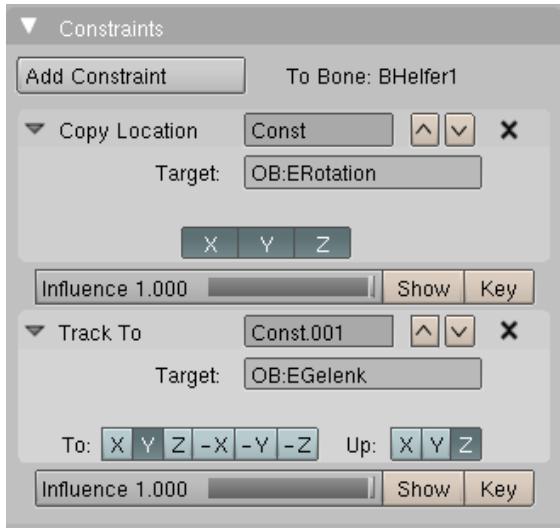


Image 5: The constraint settings for BHelper1.

of *BHelper2* governs the horizontal position of the Piston. We just have to “extract” the horizontal position, for that we’re going to use a *CopyLocation* constraint, but only in Y-direction. There is no offset for a copy location constraint - else we could have done with one empty less.

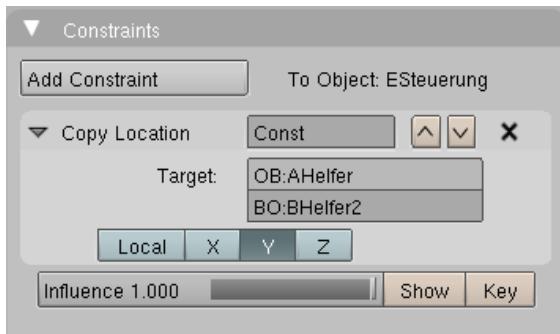


Image 6: Constraints of ESteuerung.

- *ESteuerung* gets a *CopyLocation* constraint in Y-direction to *AHelper*->*BHelper2*. You have to deactivate X and Z (**Image 6**).
- Make *ESteuerung* the parent of *AGestaenge*.

The armature *AGestaenge* gets its horizontal movement from *BHelper2* via *ESteuerung*.

Nearly finished:

- Give *BPleuelstange* a *Track To* constraint to *ERotation*.
- Parent the piston and the rod (the meshobjects) to their bones.

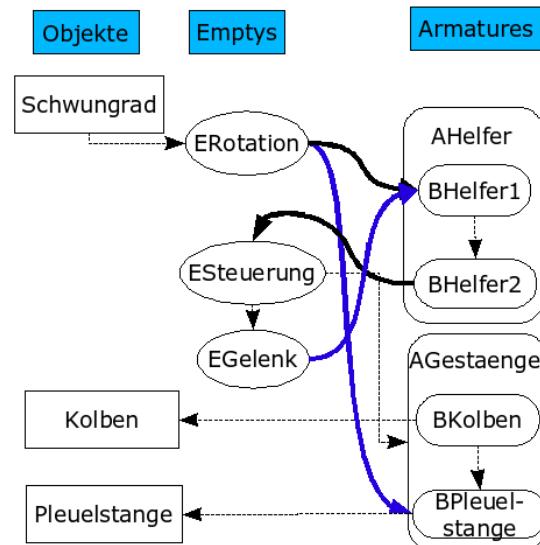


Image 7: Parent-Child Relations are drawn with slim arrows, Copy Location Constraints are drawn with thick black arrows, Track To Constraints are drawn with thick blue arrows.

Overview about the relations between armature bones and objects

- *Schwungrad* is vertex parent of *ERotation*.
- *BHelper1* has a *Copy Location X/Y/Z* constraint to *ERotation*.
- *BHelper1* has a *Track To* Constraint to *EGelenk*.
- *ESteuerung* has a *Copy Location Y* constraint to *BHelper2*.
- *ESteuerung* is parent of *AGestaenge*.
- *ESteuerung* is parent of *EGelenk*.
- *BPleuelstange* has a *Track To* constraint to *ERotation*.
- *BKolben* is parent of *Kolben*.
- *BPleuelstange* is parent of *Pleuelstange*.

The Parent-Child relations between the bones are inserted automatically through extruding, you have to insert the other relations yourself.

Finishing Touches

- Animate the rotation of the flywheel.

The driving object is the flywheel, all the other movements are synchronized with its rotation.

If you’ve done everything right, your animation should look like in **Image 8**.

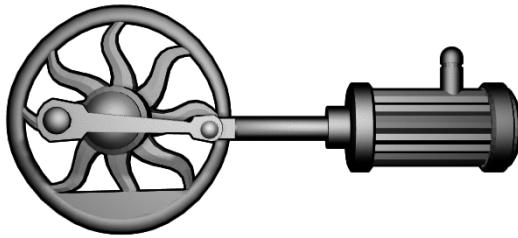


Image 8: Our working Piston, Rod and Flywheel animation.

4.67.2 Links

The German original of this tutorial (same author)

Another method using constraint system

4.68 Working example: Cutting Through Steel

In order to simulate a laser beam which is cutting through a steel plate, we have in Blender (at least) two possibilities:

1. an animated Mesh with many Shapekeys
2. an animated texture

The latter method is considerably easier to implement and the result is nevertheless convincing enough. We will proceed as follows:

- You need an animated alpha mask for the points at which the metal is changed. For this use a text object as a path. A cube follows the path and emits a particle system. The results are rendered as animation and the images stored.
- Then load these pictures and use them as an animated texture on a plane.
- With a second particle system the sparks are simulated.
- With a third particle system we simulate the annealing of the weld. Again, this sequence is used as an animated texture.
- Smoke is created with a fourth particle system.

The result is shown in **Fig. 1.**

This guide is based on the well-known tutorial [Cutting Through Steel](#), but remade with the current Blender particle system and with constraints. You need to have basic knowledge of Blender.

4.68.1 Creating the Alpha-Map

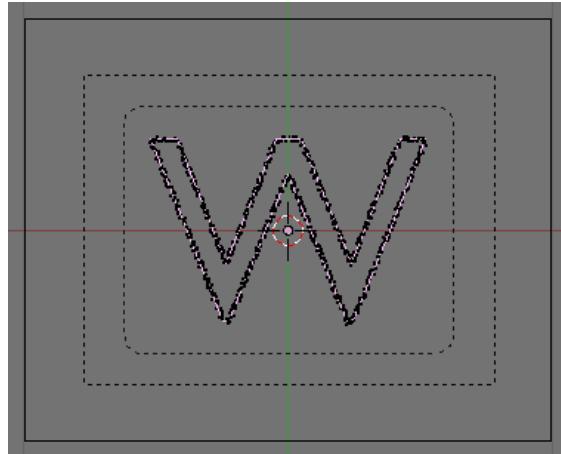


Figure 2a: Setup of curve, emitter and camera in the 3D window.

- Start Blender. Save your file. Delete the cube.

We should save the file right at the beginning, because we will save the animated textures in subdirectories to the current directory. Particle caches are also created as sub-directories.

- Set the *End* value to 150 in the *Anim* panel of the *Render* buttons. This allows you to specify the total length of the animation.
- In the *Output* panel use "`//AlphaMap/`" as output directory. When we render the animation the images are than saved in the subdirectory "AlphaMap" to the current directory.

The animated alpha map consists of gray-scale images whose luminance information is used to calculate the mask. Bright pixels produce transparency, dark pixels remain opaque.

- Use *PNG* as output format. This avoids compression artifacts which occur with the *JPEG* format.

I've left the resolution at 800x600, but you need to set your desired resolution here.

- Create the shape you want to cut out as Curve object, or use a text object and convert this to a curve. I've done the latter, I've used a text-object (a "W"), and converted this to a Curve object (*Object-> Convert Object Type ...-> Curve*).

- With the function *Center* in the *Curve and Surface* panel I've brought the object in the middle of the 3D-coordinate space.
- The filling of the object is not needed here, therefore activate the option *3D* for the curve object, even though the object should remain two-dimensional.

Now we need again a cube. The cube will follow the curve. Its particle system will create a glowing trail, that is later used as alpha map.

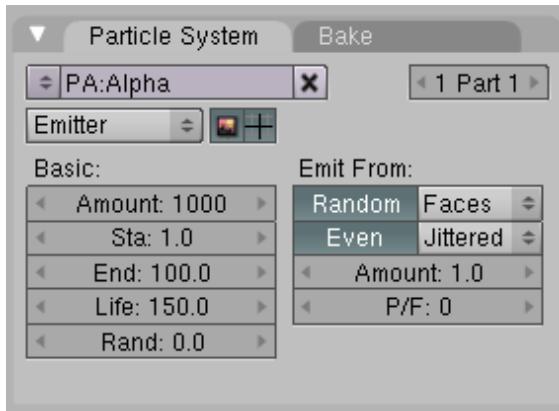


Figure 2b: particle system of the cube.

- Insert a cube.
- Scale the cube as small as the track of the laser beam (the welder) shall become.
- Select the cube, than also select the curve object with **Shift-RMB**.
- Ctrl-P->Path Constraint** creates a *Follow Path* constraint.
- Select only the cube and activate the option *Curve-Follow* in the *Constraints* panel in the *Object* buttons.

If you press **Alt-A** now and let run the animation, the cube will follow the shape of the curve object, but not necessarily in the desired direction and not from the desired location. So some work needs to be done:

- Change to the *Edit* mode of the curve object.
- Curve->Toggle Cyclic* opens the curve object, by adding (*Subdivide*) of one or more control points you achieve the desired movement. **W->Switch Direction** reverses the direction of the motion of the cube.

By adding a *Time* Ipocurve to the curve you could set the duration and speed of the animation, but I've left the preset duration of 100 Frames. The last 50 Frames the path will glow a bit and the smoke will vanish.

- Add a particle system to the cube, call this *Alpha*.
- Set the particle lifetime to 150 (or higher) (**Fig. 2b**). The particles will live the entire length of the animation.
- Bake* the particle animation.
- The cube gets a material:
 - Halo*
 - RGB 1/1/1, that is completely white
 - XAlpha*
 - HaloSize*: 0.03
 - Hardness*: 40

I want to achieve with these settings, that we will get a grayscale image, that is completely white where the material shall become transparent. We don't need partial transparency, since a relatively sharp cutting edge shall be created. Slight irregularities in the periphery are wanted.



Figure 2c: A frame of the AlphaMap.

To render the animation, the camera is centered directly over the object.

- View->Top*
- Strg-Alt-Num0** sets the camera to the current view. Select the camera.
- With the *Transform Properties* panel in the 3D-window (*Object->Transform Properties*) set the coordinates of the camera exactly vertically above the center of the 3D window (X/Y=0).
- In the *Editing* buttons of the camera set the option *Orthographic* in the *Camera* panel. Scale the size of the animation with the *Scale* value.
- The the world color to black in the *World* buttons.
- Render the animation.

Blender now generates an image sequence from 1 to 150 in the subfolder *AlphaMap* (**Fig. 2c**).

4.68.2 Welding sparks

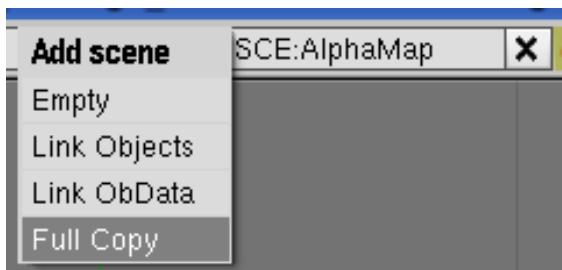


Figure 3a: Adding a new Scene

Now we'll create a new scene, in which the different effects are joined.

- Rename your current scene to *AlphaMap*.
- Create a new scene as *FullCopy* (**Img. 3a**). Name this new scene *Cutter*.

The welding sparks are created using a particle system.

- Add a *UVSphere* with 16 segments and 16 rings.
- Delete the lower half of the sphere in Edit Mode and scale it to approximately the same size as the cube.
- Switch to Object Mode.
- Select the cube in addition to the sphere. Copy the *Follow-Path*-Constraint from the cube to the sphere.

Note that you copy attributes from the active to the selected objects. The active object is the one you've selected last

Noob Note: Using Blender 2.65, the easiest way I found to do this was to use the '3D View: Copy Attributes Menu' addon. (To enable the addon press **Ctrl+Alt+U** to bring up User Preferences, click the 'Addons' tab, type 'copy' into the search box, located at the top left of the tab, and put a check in the box to the right of the addon that's found, '3D View: Copy Attributes Menu'.) First RMB click the sphere, then Shift+RMB click the cube, the one with the follow path constraint. With the 2 selected press **Ctrl+C** and choose 'copy object constraints' from the popup menu. If the sphere gets offset in the process, just move it manually to the position of the cube or the first waypoint on the path.

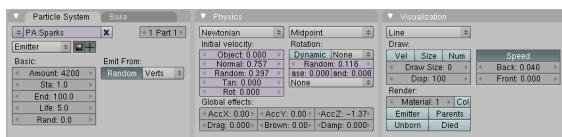


Figure 3b: The particle system for the sparks.

- Create a particle system for the sphere.

- Name *Sparks*
- *Amount:* 4200
- *Life:* 5
- *Emit From:* *Random/Verts*
- *Normal:* 0.757
- *Random:* 0.397
- *Rotation Random:* 0.116
- *AccZ:* -1.37

Except for the lifetime the values are relatively arbitrary, I have just experimented with the values until I was halfway satisfied with the results. The sparks have a slight acceleration in negative Z direction to sink down.

- *Visualization:*

- *Line* (which corresponds as much as possible to the setting *Vect* in the particle system before 2.46)
- *Speed*
- *Back:* 0.04
- *Front:* 0.0

You achieve a long spark with these settings.

- *Bake* the particles. Change to the *Bake* panel and click on the *Bake* button. The particle animation is now permanently saved.

The *Bake* end value is also the final value for the normal particle animation, even if you don't bake. If you need more than 250 frames, you need to increase this value.

4.68.3 Using the alpha map on an object

The alpha map is now used as texture on a plane. Therefore we have to position the texture exactly like the previously rendered animation, so that the sparks run in sync with the blending out of the plane.

- Add a new plane to your scene (*Add->Mesh->Plane*).
- Scale the plane at about twice its size - as big as you need it for the final shot.

The main problem is now to fit the texture exactly to the plane. We will give the plane UV coordinates, than fit the plane to the ratio of the texture and finally adjust the position in the 3D window with UV coordinates.

- Switch to *Draw Type Textured* (**Fig. 4a**).



Figure 4a: Setting the Draw Type

- Split the 3D window, and open in one part the *UV/Image Editor*.
- Switch to edit mode of the plane.
- Unwrap the plane ((*Mesh->UV Unwrap*)).
- Load an image of the alpha map in the image editor (not the first one, so you can see the course of the map).

Normally the texture is not orientated the way I would like it to be.

- Click the padlock icon in the toolbar of the *UV/Image Editor* (*Update other windows in real time*).
- Rotate the UV vertices in the *UV/Image Editor* until the texture fits (e.g., while holding the **Ctrl** key)
- Switch to object mode.

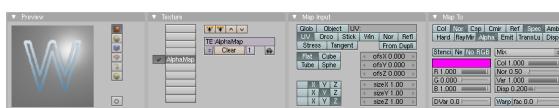


Figure 4b: Material settings for the plane

- Add a new material to the plane and name it *Rust*.
- Select the fifth texture slot and add a new texture. Name it *AlphaMap* with the parameters:
 - *Map Input: UV*
 - *Map To:*
 - *Alpha*

- *Spec*
- *Nor*
- *NoRGB*
- *DVar: 0*

With these (**Fig. 4b**) settings we achieve, that the white pixels in the texture set the Alpha and the Spec value to zero (*DVar*). Because the texture delivers RGB colors we have to activate *NoRGB*, because we need intensity values. We don't need *RayTransp*, because the world background is black anyway. We can't use *ZTransp*, because Halos are not rendered in front of materials with *ZTransp* (probably a bug in Version 2.48a). With *Nor* we get a 3D effect.



Figure 4c: Texture settings for the animated alpha map

- Switch to the *Texture* buttons.
- Type: Image Texture
- Load the first image of the alpha map sequence.
 - *Image: Sequence*
 - *Frames: 150*
 - *Map Image: Extend* (**Fig. 4c**)

Now we adjust the UV coordinates.

- *Object->Transform->Scale to Image Aspect Ratio*. This scales the ratio of the plane automatically to the aspect ratio of the image texture.
- Change to a frame where you can judge the particle animation and it's fitting to the texture.
- Switch to edit mode. Scale the UV coordinates in the UV editor until the texture fits to the particles.
- Switch back to object mode.

4.68.4 Material for the plane

We will now give the plane a better material. To judge the material we will first re-focus the camera, give it a texture and make some material settings. If you already have a good metal material, you can use this of course directly and use the Alpha texture from above on your existing material.

- Select the camera.
- Turn *Orthographic* in the *Camera* buttons off.



Figure 5: Material for the plane

- Position the camera as desired.
- Select the plane.
 - I've used the rust texture that is linked at the bottom of the page as image texture.
 - *Map Input: Orco, SizeX/Y each 0.5.* The size change is to use a larger part of the texture on the final image. But this depends on the size of the camera and the texture.
 - *Map To: Col/Nor*
 - In the second texture slot I've mapped a *Stucci* texture to *Spec*.
 - This abridged sentence means: select the texture slot. Set *Map To* to *Spec*. Deselect *Col*. Change to the *Texture* buttons. Select *Texture type Stucci*.
 - The *Specularity* is thus determined by a texture. The spec value is increased by the texture (*DVar=1*). Therefore you have to set the *Specularity* in the *Shaders* panel to the smallest value the material shall have (e.g. 0.08).

If you have done a test render, you'll see that the particle track of the cube is still there.

- We don't need the cube anymore, you may delete it. In the *Scene Cutter* we use the already generated *AlphaMap*.

Texture with OSA (*Antialiasing*) are somewhat blurred, for our purpose we use *CatRom* instead of the default *Gauss* method.

- Set in the *Render* panel *CatRom* as *AntiAliasing* filter.

We need a new output directory for the animation, to not overwrite the *AlphaMap* we've created earlier.

- Set the output directory to "://Render/" in the *Output* panel.

4.68.5 Lighting the torch

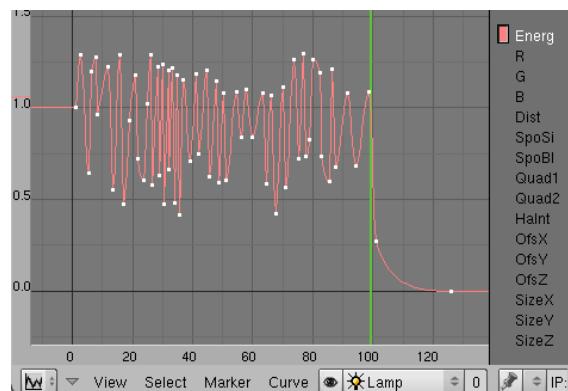


Figure 6: Ipo curve for the energy of the lamp

The torch itself shall cast light, therefore we parent a *Lamp* to the *Sphere* and animate the energy of the lamp.

- Change to frame 1.
- Select the *Sphere*.
- Place the 3D cursor on the *Sphere*. *Object->Cursor to Selection*
- *Add->Lamp->Lamp*
- Select the lamp in addition to the *Sphere*.
- *Object->Parent->Make Parent->OK*
- Select the lamp and move it a bit up (i.e. in positive z-direction). The lighting shall emit from the sparks, not from the hole.

I find it cumbersome to animate the energy of the lamp by setting ipo-keys in the lamp buttons. Instead I do it directly in the 3D window.

- Change to the *Ipo Curve Editor* in one of the open windows.
- Change the ipo type to *Lamp*.
- Activate the *Energy* channel with **LMB** click in the ipo window.
- By holding **Ctrl** key pressed, you create by left-clicking in the ipo window the points of the ipo curve.

These values may be larger than 1. However, the last maximum value should be reached in frame 100, and decrease significantly to frame 105. The sparks are only issued up to frame 100, and live up to the 105th frame. After that the metal is glowing a bit.

Your result may look somewhat like mine in **Fig. 6**. If the amplitude of the curve is not large enough, you can scale the ipo curve at the y-axis.

4.68.6 More realistic sparks



Figure 7: Better sparks

You may animate the material for *Line* visualization during the lifetime of each individual particle if you use *Halo* for the material, so you could change the sparks color from white to red.

We will show this in the next step, but confine ourself here to make a better, not animated material (**Fig. 7**).

- Add a material to the *Sphere*, name it *Sparks*.
- Activate *Halo* in the *Links and Pipelines* panel.
- Set the RGB color for *Halo* to white.
- Activate *Lines/X Alpha/Shaded* in the *Shaders* panel.
- *Halo Size*: 0.1
- *Add*: 0.8

4.68.7 Smoke



Figure 8a: Sparks and Smoke

Now were going to create an individual material animation for every particle during its own lifetime.

We will use particles with a simple texture and animate the alpha value of the material. We're going to duplicate the *Sphere* and give it a new particle system.

- Duplicate the *Sphere* with *Object->Duplicate*. The new sphere already bears the *Path* constraint.
- Remove the *Sparks* system from the new *Sphere*.
- Create a new particle system. Name it *Smoke*.
- *Amount*: 300
- *Emit from*: *Random/Verts*
- *Initial velocity*:
 - *Object*: 0.02
 - *Normal*: 0.1
 - *Random*: 0.01
- *AccZ*: 0.05
- Bake this particle system.
- Change to the *Material* buttons.
- Create a new material and call it *Smoke*.
- *Alpha*: 0.034. This is the basic visibility for the smoke texture.
- Add an ipo key for *Alpha* in frame 1.
- Change to frame 100.
- Set *Alpha* to 0.0 and add the second ipo key.

Now the smoke will disappear within the lifetime of each particle, independently of the lifetime of the particle.

- *Halo*
 - *Halo Size*: 0.224
 - *Hard*: 12
 - *Add*: 0.014
 - *Halo Tex* This is the crucial setting to use a texture for the shape of the halo.
 - *X Alpha*
- Create a new texture, you may also call it *Smoke*.
- *Texture Type*: *Image*
- Load an image with random, smoke-like structures, like that from **Fig. 8b**.

If the fading goes to quickly, or the texture is to transparent / opaque, change the animation of the *Alpha* value.

You see, even though we use only a few objects and materials, it is extremly helpful if we designate meaningfull names to the objects, materials, textures etc.

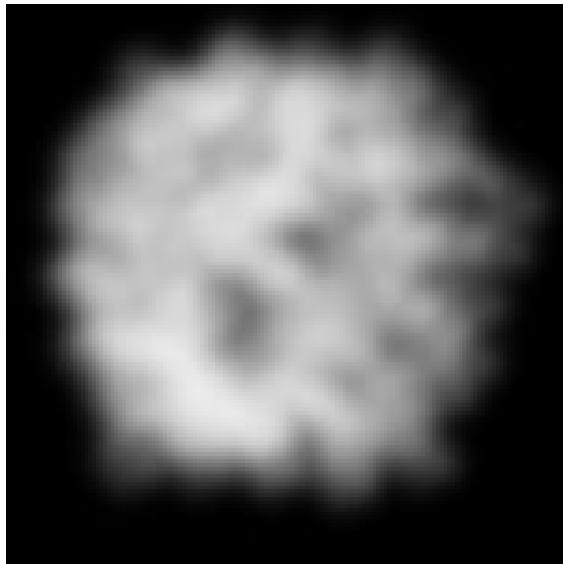


Figure 8b: Particle texture

4.68.8 After Glow

Now at last we will create another animated texture, that shall simulate the annealing of the steel plate after it is being cut through.

We use the existing particle system, but change its material settings.

- Call the material of the cube *Glow*.
- Increase the *HaloSize* to 0.04. Our new texture has to be larger than the cutout of the plane, else it will not be visible.

Now we're going to animate the material of the halos. Again this works only because we have *Point* visualization of the particles.

- Change to frame 1.
- Set an IPO key for the RGB color (here white).
- Go to 3rd Frame.
- Change the Halo color to 1/1/0.42 (light yellow). Set the next IPO key.
- Go to 6th Frame. Change the Halo color to 1/0/0 (red). Set the next IPO key.
- Go to 11th Frame. Change the Halo color to 0/0/0 (black). Set the next IPO key.

The particle trace will fade out during 11 frames.

- Render the animation.



Figure 9a: A Frame of the Glow-Map

- Change to the scene *AlphaMap*.
- Add a new scene again as a *Full Copy*.
- Name this scene *Glow*.
- Enter the output directory "*//Glow/*" in the *Output*-Panel of the *Render* buttons.

The texture shall have nice semi transparencies.

- Therefore we select a different output format in the *Format* panel, namely *RGBA* (RGB with Alpha).
- Selecting *Premul* in the *Render* panel the world background will become fully transparent, and the particle trace semi transparent.

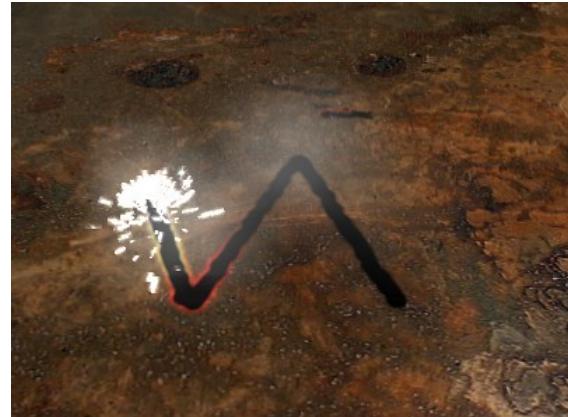


Figure 9b: A finished frame of the animation.

- Go back to the scene *Cutter*.
- Select the plane.
- Load the newly created texture in the third texture slot as an animated image texture.
 - *Add New*. Name the texture *Glow*.
 - *Map Input: UV*
 - *Map To: Col/Emit*

We need the alpha values in the image for changing the emit value. Also the alpha blends the glow texture nicely with the plane texture.

- *Texture Type: Image*

- Select the first image from the subdirectory *Glow*.
- Activate *Sequence*.
- *Frames: 150*
- *Extend*

The settings for the texture are identical with those of the texture *AlphaMap* (*Fig. 4c*), except of course for the directory in which the images reside.

In **Fig. 9b**, see a finished frame of the animation. For the video in **Fig. 1** I've changed the animation a bit, scaled the emitter objects a bit smaller, changed the sparks Color and used a different camera perspective.

Suggestions, critics and approval please write on the commentary page of this article. If you find some obvious mistakes you may correct the article directly of course.



Rust Texture

4.69 Overview

This set of tutorials is for Blender users who stumble up on the basics of certain games, such as making a HUD, menu, or even a simple button. This is also a place where users who know cool ways to spice a game up or explain troublesome aspects of making certain parts of a game genre's content work (like how to get Blender to calculate the damage of a bullet shooting).

There are a couple subsections to this. One is for those who don't like the messiness of python. Learning python to the point of being able to create a game with it will take months, maybe years, just to learn it, not to mention how long it would take to type the specific code and troubleshoot. So, the first section is for those who are much

more comfortable with using the GUI Blender screen to make the majority of their game, and the tutorials exist to help them enhance their game.

Note: No matter how much GUI you'd like to use in game creation, there is a certain amount of python you will HAVE to use to be able to really make your game good. Make sure to at least read up on it. Sorry.

Jump to *Game Creating Techniques within a GUI*

Then there is the second subsection. This section is to help those who are comfortable with python learn how to use the code in ways they never thought of, and to show those who don't do so well with it just what python can do. While it's a simple language, it can still be incredibly complicated at times, and very, very picky.

Note: DO NOT SUBMIT ANY CODE YOU DO NOT WANT COPIED A MILLION TIMES! This section is for helping the broad and the narrow range of Blender users to learn how to use python better, and there will be no expectation of attribution for submitting code you found out yourself.

Jump to *Game Creating Techniques with Python*

4.70 Advanced Game Engine Techniques (GUI)

4.70.1 Making Blender Games Graphically

If you're reading this section, I'm guessing that you have either made a game before or at least are in the process of making one right now. Let's face it: It's probably much harder than you imagined it. A large amount of people take video games and the work put into them for granted. And there are probably quite a few times in which you wanted to do something so simple—such as a menu or even a button—but didn't even know where to start or what to do.

This is why this section exists.

The next few pages will all be tutorials with different techniques on how to make objects in Blender move and respond to things like your mouse, keyboard (a.k.a. "HotKeys") and making it all look like what a real game does. They will be simple versions of a menu, or a button, in which you can use this and put it into your own idea with your own graphics and your own little tweaks.

There will be a certain amount of Python necessary in the next few tutorials, mainly just to allow the mouse icon to appear on screen. However, you can do the rest of the tutorials in the 3D View Window, and under the Logic Index in the Tab window.

4.71 Creating Pop-Up Menus

In this tutorial, we will be making a Title Screen and a Main Menu Screen to come from that Title Screen. While most people hardly pay attention to these areas on a real video game, they do make the game seem much better to play, and gives the player a certain choice of what they can do before they play the game. Before beginning this, make sure you familiarize yourself with several logic sensors and actuators, such as “Scene” and “Property”, as they will heavily be relied upon. It is also best if you have several Pictures ready to use for your game or are ready for additional functions with certain programs (a Word Processor Program and Image Editing Program [that has a transparency setting] can help move along your game.)

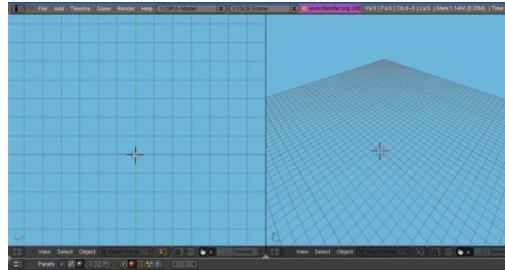
4.71.1 Making An Easy Title Menu

The Title Screen will be very simple -- it will be the “Press Enter to Continue” kind. This will lead to a more interactive kind of Main Menu, in which you can use either key commands or the mouse pointer. This tutorial will not be the most visually pleasing, but then again, you can always add whatever graphics you would like. Again, it would be wise to have handy a word-processing program and a program like Microsoft Paint. “Print Screen” will be a useful function to use while following this tutorial -- the corresponding button is generally located to the right of the F12 key in the row of function keys on your keyboard.

If you have never used different scenes before, you're about to get a crash course. Also, Blender hotkeys and actions will not be explained with (A-key) or (numpad-6); it is assumed you have already learned at least how to perform basic Blender functions.

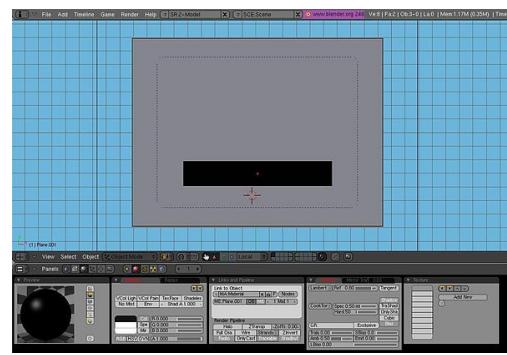
Making the Menu

First, start up Blender and delete everything in the scene. Then, move to top orthographic view (press numpad 7, or if you have not a numpad, go to the hidden top menu and in the System & OpenGL section select “emulate numpad”).



The first thing you should do is add a new scene. In the Information view panel at the top, click the SCE: dropdown menu and click on ADD NEW. Label this new scene something like “Main Menu”. Now go back to your original scene and name it something like “Title Menu”.

These names will be important when it comes to changing the scenes around. So, on your “Title Menu” Scene, begin by adding a Camera. Move this Camera in the Z direction about a point, just to get it above the grid. Now, set the camera view to Orthographic. Now switch to Camera View. Add a plane that is scaled to cover the entire surface of the Camera’s view-bounds. Move that plane just a little bit in the -Z direction. Then, add another plane, but scale this plane to be a thin rectangle about 2/3rds down the Y-length of the Camera, centered in X-direction.



(Image Shown with Materials for visual purposes, DON'T ADD MATERIALS)

Now this next step isn't necessary, but if you want a good-looking game you will probably need this step.

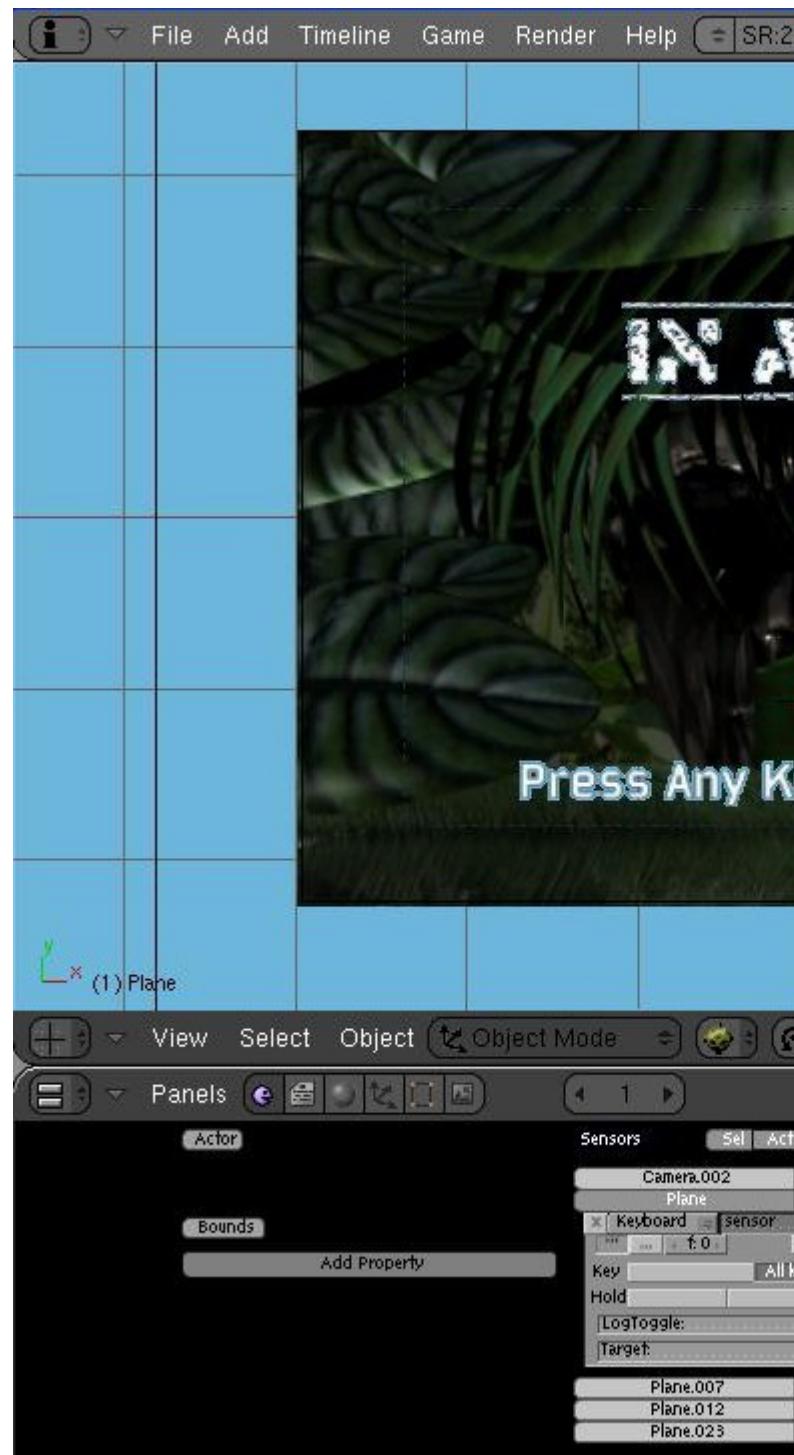
Let's add some UV textures. Open a good photo/image-editing program like Photoshop, GIMP, or Microsoft Photo Editor; something that preferably supports Transparency. Type some text appropriate for your game such as, “Press any key to continue.” What I do is add lots of extra space and add additional text that I will use in the main menu too, such as “New Game”, “Continue”, “Story Mode”, etc. so it is all in one image for multiple UV textures. If your editor supports it, set the background color to transparent. If not, set it to black so it will register as 0 on the alpha channel in Blender. Save the image (GIF or PNG are reasonable choices for use with transparency). Now, in Blender, split the 3D screen into 2 screens, and change one screen over to UV face select. Select your small rectangle, and switch to Edit Mode and Unwrap the face in the 3D window, then open your saved image in the UV face window. Scale your face in the UV window so it holds only the “Press Any Key to Continue” graphic. If you wish, also add a starting Image on the larger rectangle.

Setting Up The Actions

Now this step is necessary. Select the large background rectangle, in the Buttons Window at the bottom switch it to the Logic Tab. This will be very simple. Add one Sensor, one Actuator, and one Controller. Make the sensor a Keyboard Sensor, depress the “True Pulse” button, and in the key area depress the “All Keys” button. Make the Controller an “AND” controller. Make the Actuator a Scene Actuator. Select the Drop-Down box

and make it “Set Scene” wit “SCE:Main Menu”. **Before you test the game, make sure to save and add a Camera to the “Main Menu” Scene. Otherwise the game WILL crash.**

If you want the option of returning to this scene, instead of one Actuator make two. Make both of them Scene, but make one “Suspend Scene” with “SCE:Title Menu” and make the other Actuator “Add Overlay Scene” with “SCE:Main Menu”. Also, that means in the Main Menu Scene later you'll have to add some key to hit or some action to “Remove Scene” “SCE:Main Menu” and “Resume Scene” “SCE:Title Menu”.



4.71.2 Author's Starting Menu Example

The following contains actual in-progress work by the original Author of this page. The work and ideas are heavily copyrighted and it is highly encouraged (begged) that you do not rip this content off-the author has submitted this work generously to help less advanced users more efficiently.

The following is my work on a game I am making, and I hope it helps you too.

This will get you to the next menu, which will be on the next section.

4.71.3 A Main Menu

This menu will be more difficult, using a mouse-python function to use and make the mouse appear when you play your game and a maze of sensors and actuators entangled.

Here is the first thing you need: a python script to make the Mouse show ---

**This code is attributed to the author of the chapter “Blender Game Engine/A Simple Mouse Pointer” and this is not my own code.*

```
import Rasterizer as r
r.showMouse(1)
```

Put this in the text editor, and save the text as something simple like “Mouse”. Now, on your Main Menu scene, select the camera and add 1 sensor, 2 controllers, 1 actuator, and 1 property. Name the property “Switch” “Int” Type, and set it to 0. Set up your logic as follows:



A change will need to be made in order to make yours work. The python controller should be Script:Mouse, not Script:showpointer. If you can't see the connections:

- Connect the Sensor to the 2 controllers
- Connect the AND controller to the actuator
- Leave the Python controller open-ended

If you play the game, you will see the mouse then! Now, back to the Menu, there are multiple kinds of Menus you can make when it comes to Menus popping up. The one I believe works best is a 3-layered button. To explain what this means, I'll run you through the steps of making one. Let's start with the layers themselves. (For this to work best, make the background of any images you use have transparency, so you can make invisible-yet-clickable faces.) Create a rectangle facing the camera where you want your first menu option to be. Copy the object until you have 3 rectangles, and make each one very slightly in front of the last. Now, Select the back rectangle, and on this one apply the UV face of the option's words. The second rectangle is the UV face for a graphic outline around the words if the option is selected in the game. It's not necessary but then again, it helps the player with selecting options and it looks nice. The top rectangle will be the selectable one--make the UV face applied only over a completely transparent area. Now, for the logic.

Text-Rectangle:
-No Logic Needed

Outline-Rectangle:

-Give it a Property named “Selected”, Int Type, starting at 0.

-Make a Property Sensor, True Pulse, “Selected” Property is Equal to 1, Connected to an AND controller, Connected to a Visibility Actuator at “Visible”

-Make a Property Sensor, True Pulse, “Selected” Property is Equal to 0, Connected to an AND controller, Connected to a Visibility Actuator at “Invisible”

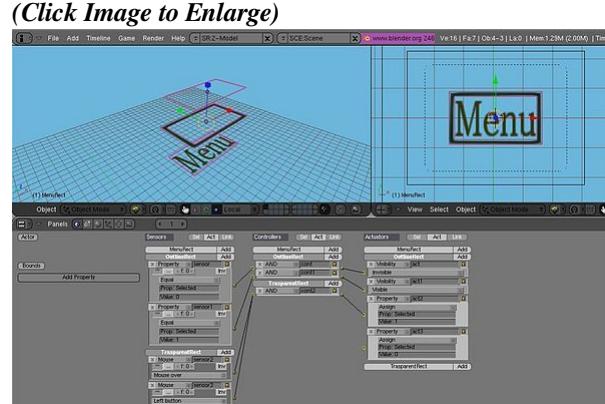
-Make a Property Actuator, “Selected” Property is Assigned Amount 1 -Make a Property Actuator, “Selected” Property is Assigned Amount 0 -Create AND Controller and connect both Mouse Sensors. AND Controller connect to the Outline-Rectangle

Actuator “Selected” Assign=1.
Script: showpointer

AND Transparent-Rectangle ForPointer
2 Mouse sensors: “Mouse Over” and “Left Click”

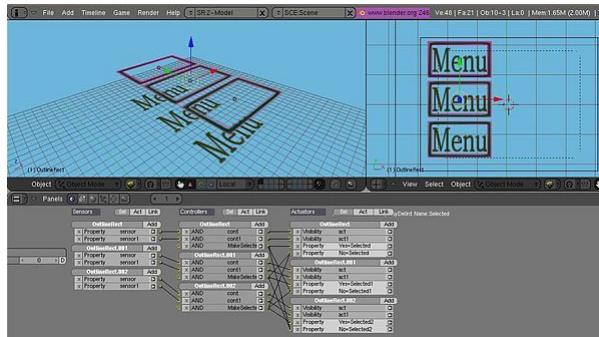
Assign Prop: switch Value: 1

Scene Add Prop



Now here is the trickier part to explain. To make things a little more simple, name that last Outline-Rectangle’s AND controller connected to the 2 mouse sensors. Select all 3 layers, and duplicate them 3 times. Make each one (in camera view) below the last, making sure to keep them parallel on the Z axis. Now, to make only one selected at a time, take the named controller of the first, and connect it to the Outline-Rectangle’s Property Actuator “Selected” Assign=0 of second and third button. Do the same from the second button to the first and third one, and same with the third to the first and second. It should look like the picture:

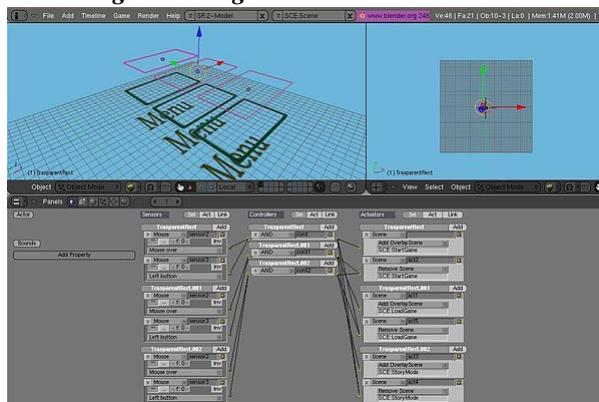
Click Image to Enlarge



This logic arrangement should yield (when the game is played) options in which if you click one it will light up, and if you click another, that one will go down and the other will light up.

Now this looks nice aesthetically, but the use of it is now all you have to do is make some Scene Actuators. Add 2 more actuators to each of the Transparent Planes. Use the same method as you did with the Properties, but instead of “Property Assign 1/0” make 1 AND controller, 1 Scene Overlay and 1 Scene Remove for each transparent button. Now make 3 additional Scenes, and either make an entirely new menu for each, or to be easier just make colored rectangles on the scene, different colors for each scene -- either way, make sure that, if the scenes overlay, the additional scene won't cover up the buttons of the Main Menu Scene. So, now connect the 2 Mouse Sensors to the corresponding AND Controller, and connect that to the Corresponding “Overlay Scene” Actuator and 2 Opposing “Remove Scene” Actuators. Do this with the other 2 Transparent Rectangles. Make sure your new scenes also have cameras and that they are scaled properly.

Click Image to Enlarge



You can use this technique to make Menus and Sub-Menus Galore! It's fairly simple to make multiple sub-levels of this and get back to the original scene with this method.

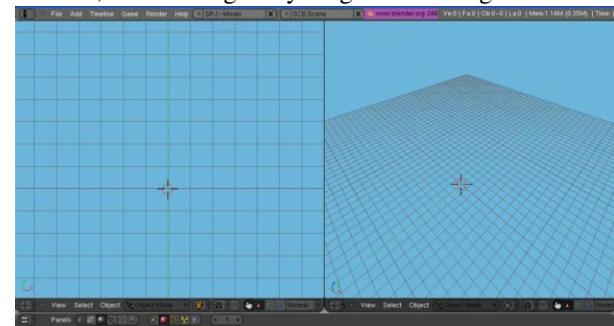
I hope this helped you make your menus work well.

4.72 Creating Dropping Menus

In this section, you are going to learn a variety of ways to make menus with motions. These menus can be used in more ways than the simple pop-Up menus, such as in-game if you need to select something you can use this to bring up a menu smoothly. In this section there is heavy use of Properties, “Property” Logic Blocks, and especially IPO usage. When you do it once, it will be very much easier to do it again.

4.72.1 Making Moving Selectors

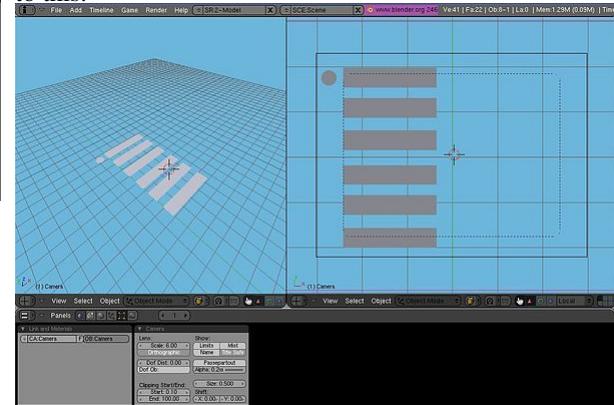
To kick this tutorial off, let's just do a simple icon that will be next to the selected menu item. There will be 6 possible menu items, which can be controlled with both the arrow keys and the mouse. Let's begin by booting up Blender, and deleting everything--then adding a camera.



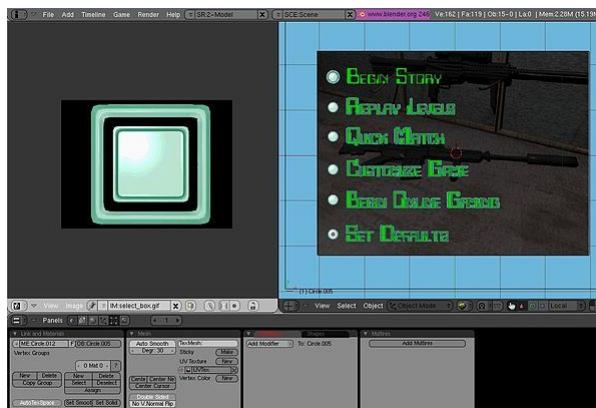
Set this camera to orthographic. Now I like making things look nice, but here are the basics of what you will need for this tutorial:

- Make 6 Planes, all different objects, and put them in a vertical list.
- Make a circle next to the first plane. This will act as the Selector.

So after you've made your objects, it should look similar to this:



For me, I made mine look more like this, a little more detailed:

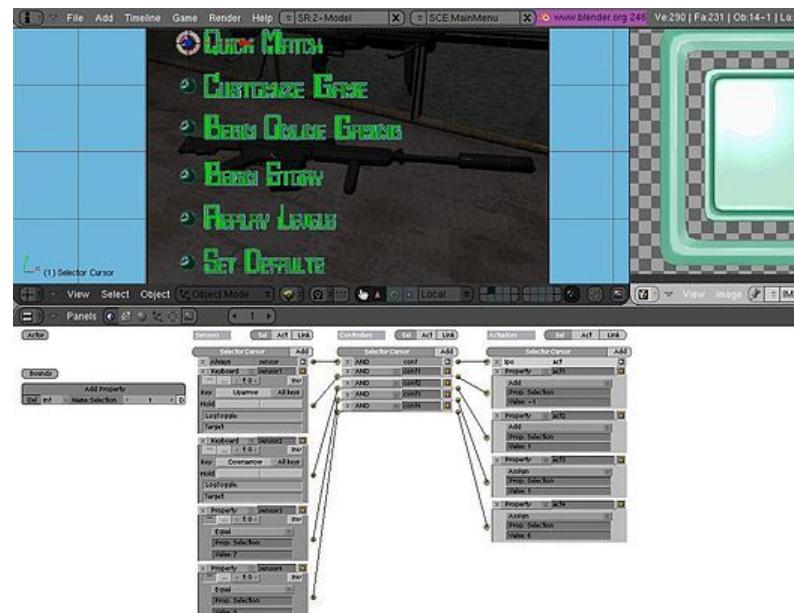


So now that we have our menu set up, it's time to set up the properties that will tell the selector how to move. Select the selector that will be moving and give it a property name "Selection". Make it "Int" type, and start it out at "1". To get this selector to move the way we want it, we have to set up its movements. Make sure you are on frame 1, and put the cursor on the first menu item, then Insert Keyframe for "Loc". Next, move to frame 2, and put the cursor on the second menu item, then insert "Loc" Keyframe, and continue in this fashion for all your menu items. To get this applied in the Game Engine, in the logic bricks, add an "Always" Sensor (True Pulse Enabled), connected to an "AND" connector, and connected to a "IPO" Actuator, set to "Property" with Prop:Selection. Now we can do 3 things with this selector to have it move like a selector in a video game:

Up/Down Arrow Keys

To make the up and down arrow keys work properly, Select the Selector and add four sensors, controllers, and actuators. Make two sensors keyboard, with one set to "UpArrow" and one set to "Downarrow" (do not activate pulse triggering). Make the other two sensors Property Sensors, with True Pulse. For the first, make it "Equal", Prop:Selection, Value:0. For the other, make it "Equal", Prop:Selection, Value:7. For the Actuators, make all four "Property" sensors. For two of them, make them "Add", Prop:Selection, and make the first "Value:-1" and the second "Value:1". The other two should be "Assign", Prop:Selection, with the first at "Value:1" and the next as "Value:6". Now connect them as follows:

Click To Enlarge Image



```

"Keyboard:UpArrow\char"0022\relax{ }--\char"0022\
relax{ }AND\char"0022\relax{ }--\char"0022\
relax{ }Prop:Selection/Add/Value:$-1"
"Keyboard:Downarrow\char"0022\relax{ }--\
char"0022\relax{ }AND\char"0022\relax{ }--\
char"0022\relax{ }Prop:Selection/Add/Value:1"
"Prop:Selection/Equal/7\char"0022\relax{ }--\
char"0022\relax{ }AND\char"0022\relax{ }--\
char"0022\relax{ }Prop:Selection/Assign/Value:1"
"Prop:Selection/Equal/0\char"0022\relax{ }--\
char"0022\relax{ }AND\char"0022\relax{ }--\
char"0022\relax{ }Prop:Selection/Assign/Value:6"

```

Now you can play your game, and the arrow keys should work just fine!

HotKeys

If you don't know what a HotKey, it's a way to access menu items or action by pressing a specified letter, instead of just clicking or arrow keys. This method will require a different step when you make your UV text textures, as you will need to somehow change one letter (either the color or possibly underline) to let people know that that is a HotKey action.

The HotKey system is easy to set up right alongside the IPO/Selection property. In each menu item, highlight or make special one unique letter in each menu item. Then, with the selector selected, set up 6 keyboard logic sensors, each one for a different letter. Add 6 controllers, and 6 actuators as well. Make all the actuators "Property", "Assign", Prop:Selection, starting with Value:1 progressing to Value:6"

Mouse Selection

Our final selection method we can add is the mouse scroll over. Set up the Mouse Sensor to Scroll Over for all 6 invisible selection planes. Then, make the AND

Controllers and connect to your original set of “Set Property” Actuators. that matches the Object’s IPO property number. It’s fairly simple.

Make the Selection

The final thing to do is, once the selections are set up, make another 8 sensors, 7 controllers, 6 and actuators. Now make the 7th one a keyboard sensor for the ENTER key, as most people usually think they can start a game by pressing ENTER. The 8th Sensor is for another Mouse Sensor for “Left Click” so that if any of them is highlighted and you Left-Click the mouse, then it will be equivalent to pressing the ENTER key. That may be confusing as, unless you set up the dozen extra sensors, controllers, and actuators, you don’t need to scroll over the selection to pick it, so that 8th sensor is your choice whether you want to add it. Connect the 7th and 8th sensor to the 7th controller. If you have the 8th sensor, make it an OR controller; otherwise, make the controller AND. As for the first 6 sensors, they should be Property Sensors for "Prop:Selection" "Value:[1-6]". Connect each to their own separate controller. Now connect the 7th controller to all 6 Actuators, and the first 6 controllers connect to their own Actuator alongside the 7th. The Actuators can be whatever you want them to: whatever the selection causes to happen when you select the item selection.

4.72.2 Make the Menu Move

So we created a selection-based moving icon on a menu, but what might happen if you select that item and need a sub-menu in-game? You can't very well pause the game every time (unless it's a pause menu). This is where making the menu itself move comes in very handy. In this section, you will make an object clickable, and once clicked it will give out a submenu, which in itself will be able to give out a submenu.

The IPOS

4.73 The “5-Layer” Button

4.73.1 The 5-Layer Button

4.74 Creating Object Outlines

4.74.1 Creating Object Outlines

4.75 Advanced Game Engine Techniques (Python)

4.75.1 Game Creation Techniques (Python)

4.75.2 Additional Resources

- Blender 3D: Blending Into Python
- GLSL Programming in Blender

4.75.3 External Links

- <http://www.theboomshelter.com/hss.html>
- <http://www.youtube.com/watch?v=7hbCEyyVYHI>

4.76 Hacking Blender

Blender is an **Open Source** project. That doesn’t just mean you get to use it for **free**, you also get to see how it works, and you can even make your own changes and share them with others.

Blender is also a large software project (well over a million lines of code), with a great many active contributors over a lifespan of more than a decade, and it continues to be developed at a rapid rate. This can make things somewhat intimidating for less-experienced programmers.

This unit assumes you have some decent programming experience under your belt. Blender is mainly programmed in the **C**, **C++** and **Python** programming languages. It can be built using either the **CMake** or **SCons** build systems.

4.76.1 Getting the Blender Source Code

The official Blender source is kept in **Git** repositories located at developer.blender.org. There are actually several separate repositories:

- `blender-main` — the main part of the Blender source, excluding most Python addons.

- `blender-addons` — the Python addons included in the standard Blender distribution.
- `blender-addons-contrib` — additional useful Python addons.
- `blender-translations` — localized language translations for text messages.
-
- `blender-tests` — some interesting example .blend files used for testing and demonstrating Blender functionality.
- `blender-dev-tools` — tools that are useful for performing maintenance tasks on the Blender source, but are not actually needed for building Blender.
- `blender-cloud` — looks like a framework for offering a new cloud-based Blender service.

4.76.2 Layout of the Blender Source

Say you've checked out a copy of the main Blender source tree. The top level looks like this:

- `build_files/` — files used during the build process
- `CMakeLists.txt` — top-level control file for CMake
- `COPYING` — note about Blender licensing (GPLv2)
- `doc/` — documentation files, among them:
 - `doc/blender_file_format/` — the format of .blend files
 - `doc/build_systems/` — how to build Blender, and hack the build system
 - `doc/manpage/` — the man page
- `extern/` — non-Blender-specific libraries, primarily developed elsewhere, included in the source
- `GNUmakefile` — simple build script for those who can't be bothered to go through the CMake setup process
- `intern/` — libraries which are non-Blender-specific but primarily developed here, also glue code for interfacing to external libraries not included in the Blender source. Notable subdirectories:
 - `intern/bsp/` — Constructive Solid Geometry routines
 - `intern/cycles/` — the Cycles renderer
 - `intern/elbeem/` — the fluid simulator

- `intern/ghost/` — Blender's platform-independent GUI (including platform-dependent implementations). See `intern/ghost/GHOST_ISystem.h` for a more detailed description.
- `intern/guardedalloc/` — thread-safe memory management with consistency checking
- `intern/iksolver/` — the Inverse Kinematics library
- `intern/smoke/` — the library for simulating smoke and flames
- `release/` — additional files to be included in the Blender distribution, including GUI icons and fonts
- `SConstruct` — top-level control file for SCons
- `source/` — the main part of the Blender source, further divided (ignoring the `CMakeLists.txt` and `SConscript` files which you will find just about everywhere) into
 - `source/blender/` — the bulk of the source, of which some useful parts are
 - `source/blender/blenlib/` — low-level stuff for file management, geometry algorithms, sorting and suchlike
 - `source/blender/blenkernel/` — core Blender-specific code (no UI stuff)
 - `source/blender/blenloader/` — code for reading and writing .blend files
 - `source/blenderplayer/` — some additional files for building the Blender Player executable
 - `source/creator/` — the Blender mainline
 - `source/gameengine/` — the Blender Game Engine
 - `source/icons/` — some files specific to the Windows build
 - `source/tests/` — some test scripts

Common Subdirectory Layout

Within many of the subdirectories in `source/blender/` and `intern/`, you will see the following pattern:

- A bunch of .h files in the directory, and
- an intern subdirectory.

The .h files define the functionality exported to other modules, while the intern subdirectory contains the .c or .cpp files that actually implement the module. Sometimes the .h files are put into an extern subdirectory instead of the upper directory level.

(And yes, these meanings of intern and extern are not the same as the meanings of intern and extern at the top directory level.)

4.76.3 Blender’s “Genetic Code”: “DNA” and “RNA”

You will find references to “DNA” (or “SDNA”) and “RNA” throughout Blender’s source code. The analogy to the terms from genetics is weak at best, but what they refer to is:

- “DNA” or “SDNA” (“structure” DNA?) is the system for mapping Blender’s in-memory data structures to the on-disk .blend file format. A .blend file is little more than a memory dump, which makes it quick to write. However, the file needs to be readable by Blender builds on machines with different endianness, 32 versus 64-bit pointers, etc, not to mention future versions with different capabilities. Thus, the saved .blend file includes a detailed description of the layout of all the data structures saved in the file, and this description is stored in the “DNA1” block at the end. This block is generated by the makesdna tool, which is built and run automatically as part of the overall build process, so its output can be included directly into the Blender executable. It parses all the .h files in the directory source/blender/makesdna/, so all data structures that could get saved into .blend files must be defined here, and they must be careful to use a limited subset of C that the parsing tool can handle.
- “RNA” defines the Python interface to Blender’s internal data structures and routine calls.

Here is Ton “Mr Blender” Roosendaal explaining DNA and RNA.

4.76.4 Special Globals: “G” and “U”

There is a frequently-referenced global variable named “G”, of type struct Global, declared in source/blender/blenkernel/BKE_global.h. The variable is defined in source/blender/blenkernel/intern/blender.c. This same file also defines the global “U”, of type struct UserDef, declared in source/blender/makesdna/DNA_userdef_types.h.

4.76.5 Naming Conventions

Some (but not all) global symbols have prefixes on their names indicating (roughly) where they come from. Here is a partial list.

4.76.6 User-Interface Implementation

The UI code is structured into several layers. This code also handles finding of user-preference files and shared application data. Starting from the lowest, the layers are:

- intern/ghost/
- source/blender/windowmanager/ (see source/blender/windowmanager/WM_types.h for an overview)
- source/blender/editors/screen/
- source/blender/editors/interface/

4.76.7 See Also

- Building Blender — platform-specific instructions on the Blender Wiki
- Development resources at blender.org, including
 - Architecture overview
 - Official developer blog
- Mailing lists

4.77 Introduction to Game Engine Source

Mission Statement: The purpose of this section is to show a programmer how to make sense of the blender game engine source code. It is not intended for those who cannot code in C++. If you’re one of the people who have a basic understanding, I’m being careful to use keywords that if you google, should provide enough information to understand.

When you press P, several things are sent to the GE before anything can be done. What might these be?

```
StartKetsjiShell(struct ScrArea *area, char* scenename,
struct Main* maggie1, struct SpaceIpo *sipo, int always_use_expand_framing)
```

Blender uses a lot of structs and classes. You need to have a working understanding of these as well as inheritance before we can move on. For a refresher: http://www-numi.fnal.gov/offline_software/srt_public_context/WebDocs/Companion/cxx_crib/inheritance.html

The SCA_IObject class
 source/gameengine/GameLogic/SCA_IObject.cpp
 and SCA_IObject.h

This class encapsulates a lot of other things, so we can start here and move downwards to the things that it controls. First off it uses vectors to store all the different Sensors, Controllers and Actuators that are linked to the object. *A vector is like an array, but it is dynamic and has built in functions.*

We will begin by examining how sensors are added.

```
typedef std::vector<SCA_ISensor *> SCA_SensorList;
SCA_SensorList m_sensors;
```

The first line here is a typedef so it just tells us that we can create a vector of SCA_ISensor by using the name SCA_Sensor list.

The second line creates the vector with the name, m_sensors

```
void SCA_IObject::AddSensor(SCA_ISensor* act)<br>/> { m_sensors.push_back(act); }
```

This function allows the parents of an SCA_IObject to add new sensors by calling: SCA_IObject.AddSensor(keyboard); *pseudocode*

So now you know how Objects in the GE create sensors (and actuators and controllers, its the same :) So lets look at how Sensors are grabbed! Since they are dynamically added the program can't make assumptions about how many are there or what they will be. Thus there is this helper function:

```
1"SCA_ISensor*      SCA_IObject::FindSensor(const
STR_String& sensorname) 2{ 3 SCA_ISensor* found-
sensor = NULL; 4 5 for (SCA_SensorList::iterator its
= m_sensors.begin();!(its==m_sensors.end());its++) 6
{ 7 if ((*its)->GetName() == sensorname) 8 {<br />
9 foundsensor = (*its); 10 break; 11 } 12 } 13 return
foundsensor; 14}"
```

Put simply, you give this the name of a sensor and it returns a pointer to the sensor.

for the newbies: I know you learned all about classes when i asked you to, but just in case:) the first word, "SCA_ISensor*" this tells us that the function will return a pointer to a SCA_ISensor. And based on the functions arguments, we know that when you call FindSensor you must give it a string between the parenthesis.

in line 3 you create an empty pointer which you will use to store the sensor that you find. Then you loop and go through each sensor in the vector of sensors (remember, it was named m_sensors) until you find the one that matches the name. If you don't find one, you return NULL.

void SCA_IObject::Suspend(void) What this does is it loops through all of the sensors and sends each one a "suspend" call. What this does is change a variable named, "m_suspended" which makes stops the sensors from running when they are called with their Activate() function.

Chapter 5

Appendices

5.1 Glossary

5.1.1 A

- **Alpha Channel** is an additional channel in a 2D image for transparency. In an image element which stores a color for each pixel, an additional value is stored in the alpha channel containing a value between 0 and 1. A value of 0 means that the pixel does not have any coverage information; i.e. there was no color contribution from any geometry because the geometry did not overlap this pixel. A value of 1 means that the pixel is fully opaque because the geometry completely overlapped the pixel.
- **Ambient Light** is light that doesn't seem to come from a specific source, but is just there. Look under the desk - it's pretty dark, but there's some light there. In the real world, this is caused by stray photons bouncing around and occasionally ricocheting under the desk. Ambient light is the basic, minimal amount of light in the whole scene. Adding too much ambient light makes a scene look washed out. Since the light doesn't come from anywhere, all sides of an object are illuminated equally, and it won't have any shading on it.
- **Ambient Occlusion (AO)** is a ratio of how much ambient light a surface point would be likely to receive. It simulates a huge dome light surrounding the entire scene. If a surface point is under a foot or table, it will end up much darker than the top of someone's head or the tabletop.
- **Armature** is the interconnection of bones that form the skeleton of an animated figure. The Inverse Kinematics library contains the code to make armatures move. The armature must still be **rigged** with 3D objects to give shape to its head, hands, trunk, feet, etc.

5.1.2 B

- **Background image:** a 2D image (“picture”) that is placed “behind” the entire 3D scene, like a backdrop on a movie set. Blender permits the placement of these images in all six directions from the origin: back, front, top, bottom, left, right.
- **Bake:** to precompute computationally-intensive elements of an animation. For example, in a physics simulation involving the behaviour of fluids or clothing, you would set up the physical parameters, then compute (*bake*) the positions and shapes of the objects over the duration of the animation. Afterwards, you can assign materials and lighting, and then render the frames to produce the actual animation. Doing the baking as a separate step, and saving the results from that, means you can change your mind about the materials and lighting and rerender the frames more quickly.
- **Bézier surfaces** were first described in 1972 by the French engineer Pierre Bézier who used them to design automobile bodies. Bézier surfaces can be of any degree, but bicubic Bézier surfaces generally provide enough degrees of freedom for most applications.
- **BF** is Blender Foundation
- **Blend** - *to Blend*, working with Blender; also Blender's file extension.
- **Bounce Light:** Simple lighting situations have a single light, called a key light, illuminating one side of an object. This creates strong shading and definition of the volume of the object. However, a 3D light will often make the contrast too great - the dark side of the object is completely black since no light is hitting it. In reality it would still be lit a little, just not as much as the brightly lit side, because of light bouncing around the room and hitting the dark side of the object. In realtime 3D, bounce light is not calculated, so you have to create it yourself. Either add

a little ambient color, or put a second, less bright directional light pointing the opposite direction to give a little light to the shadows.

- **Bump mapping** is a technique where at each pixel, a perturbation to the surface normal of the object being rendered is looked up in a texture map and applied before the illumination calculation is done. Bump Mapping uses a gray-scale image map to change the direction of surface normals. You can use this to simulate height, so that you can paint wrinkles and bumps. 50 % grey means neutral (no change is made), lighter means higher, darker means lower. Note that the position of faces is not actually changed; by rotating just the normals, lighting will change too, to give the illusion of a height difference. This has downsides too: the outline of objects isn't changed, so the trick is given away. For similar effects you can use Displacement Mapping and Normal Mapping.

5.1.3 C

- **Caustics** in optics is a bundle of light rays. For example a caustic effect may be seen when light refracts or reflects through some refractive or reflective material, to create a more focused, stronger light on the final location. Such amplification, especially of sunlight, can burn — hence the name. A common situation when caustics are visible is when some light points on glass. There is a shadow behind the glass, but also there is a stronger light spot. Nowadays, almost every advanced rendering system supports caustics. Some of them even support volumetric caustics. This is accomplished by raytracing the possible paths of the light beam through the glass, accounting for the refraction, reflection, etc.

5.1.4 D

- **Depth of Field** (DOF) is the distance in front of and behind the subject which appears to be in focus. For any given lens setting, there is only one distance at which a subject is precisely in focus, but focus falls off gradually on either side of that distance, so there is a region in which the blurring is tolerable. This region is greater behind the point of focus than it is in front, as the angle of the light rays change more rapidly; they approach being parallel with increasing distance.
- **Diffuse Light** is even, directed light coming off a surface. For most things, the diffuse light is the main lighting we see. Diffuse light comes from a specific direction or location, and creates shading. Surfaces facing towards the light source will be brighter,

while surfaces facing away from the light source will be darker.

- **Directional Light** is a light that has a specific direction, but no location. It seems to come from an infinitely far away source, like the sun. Surfaces facing the light are illuminated more than surfaces facing away, but their location doesn't matter. A Directional Light illuminates all objects in the scene, no matter where they are.
- **Displacement Mapping** uses a greyscale heightmap, like Bump Mapping, but the image is used to physically move the vertices of the mesh at render time. This is of course only useful if the mesh has large amounts of vertices, but the (relatively) new “Simple Subdiv” subsurf option allows you to add more vertices at render time which will be moved by the displacement. This makes it much slower than Bump Mapping, as there need to be many more faces to render, but it is much more realistic.

5.1.5 E

- **Environment Maps** (EnvMaps) is the method of calculating reflections. Involved rendering images at strategic positions and applying them as textures to the mirror. Now in most cases obsoleted by Raytracing, which though slower is easier to use and more accurate.

5.1.6 F

- **Focal Length** of a lens is the distance along the optical axis from the lens to the focus (or focal point). The inverse of a lens' focal length is called its power.
- **Focus** of a lens is the point onto which collimated light parallel to the axis is focused.
- **Foreshortening**
- **Fresnel** lens is a type of lens invented by Augustin-Jean Fresnel. Originally developed for lighthouses, the design enables the construction of lenses of large size and short focal length without the weight and volume of material which would be required in a lens of conventional design. As it relates to rendering, fresnel refers to the tendency of materials to be more reflective when light strikes at a high angle of incidence-- think of how sunlight reflects from distant water, but penetrates closer water, or of how road glare is most extreme at dawn or dusk. This varies with material, and specification of fresnel is an important part of material definition.

5.1.7 G

- **GE** is Game Engine.
- **Global Illumination** (GI) is a superset of radiosity and ray tracing. The goal is to compute all possible light interactions in a given scene, and thus obtain a truly photorealistic image. All combinations of diffuse and specular reflections and transmissions must be accounted for. Effects such as colour bleeding and caustics must be included in a global illumination simulation.
- **Gouraud shading** is a method used in computer graphics to simulate the differing effects of light and colour across the surface of an object. In practice, Gouraud shading is used to achieve smooth lighting on low-polygon surfaces without the heavy computational requirements of calculating lighting for each pixel. The technique was first presented by Henri Gouraud in 1971.

5.1.8 H

- **High Dynamic Range Image** (HDRI) is a set of techniques that allow a far greater dynamic range of exposures than normal digital imaging techniques. The intention is to accurately represent the wide range of intensity levels found in real scenes, ranging from direct sunlight to the deepest shadows. The use of high dynamic range imaging in computer graphics has been popularised by the work of Paul Debevec. Blender uses Yafray for these techniques.

5.1.9 I

- **Index Of Refraction** (IOR) is about the way that light passes through different types of materials... diamond, glass, water etc. When a light ray travels through the same volume it follows a straight path. However if it passes from one transparent volume to another, it bends. This is why a straw in water looks bent. The amount of bending differs between materials. The angle by which the ray is bent can be determined by knowing two things: the angle at which the incoming ray has been cast and the Index of Refraction. This IOR value is unique for every material. Glass has an IOR of about 1.5 and water 1.3. By increasing the IOR value for a Blender material, you can control how much the environment behind the transparent object is distorted, and thus improving the realism of the shader.
- **Interpolation** (IPO) is an animation curve: it indicates how the object must “move” between an initial and a final position, at the rate determined by the

rendering engine. Objects can be animated in many ways. They can be animated as Objects, changing their position, orientation or size in time; they can be animated by deforming them; that is animating their vertices or control points; or they can be animated via very complex and flexible interaction with a special kind of object: the Armature.

- **Inverse Kinematics** (IK) is the process of determining the movement of interconnected segments of a body or model, starting from the desired motion of the endpoints of the armature. Using ordinary Kinematics on a hierarchically structured object you can for example move the shoulder of a puppet. The upper and lower arm and hand will automatically follow that movement. IK will allow you to move the hand and let the lower and upper arm go along with the movement. Without IK the hand would come off the model and would move independently in space. The Blender Armature System includes Inverse Kinematics. For general armatures there are many possible solutions to the IK.

5.1.10 J

- **JPEG** Acronym for Joint Photographic Expert Group (pronounced jay-peg) is a commonly used standard method of lossy compression for photographic images. The file format which employs this compression is commonly also called JPEG; the most common file extensions for this format are .jpeg, .jfif, .jpg, .JPG, or .JPE although .jpg is the most common on all platforms.

5.1.11 K

- **Keyframe** is a frame in an animated sequence of frames that was drawn or otherwise constructed directly by the user. When all frames were drawn by animators, the senior artist would draw these frames, leaving the “in between” frames to an apprentice. Now, the animator creates only the first and last frames of a simple sequence; the computer fills in the gap. This is called tweening.

5.1.12 L

- **Luminosity** (more properly called luminance) is the density of luminous intensity in a given direction. In astronomy, luminosity is the amount of energy a body radiates per unit time. It is typically expressed in the SI units watts, in the cgs units ergs per second, or in terms of solar luminosities, L_s ; that is, how many times more energy the object radiates than the Sun, whose luminosity is 3.827×10^{26} W.

5.1.13 M

- **Motion Blur** is the simulation of the phenomenon that occurs when we perceive a rapidly moving object. The object appears to be blurred because of our persistence of vision. Doing motion blur makes computer animation appear more realistic. It can be thought of as adding back some of the time dependence expressed in the Rendering Equation.

5.1.14 N

- **Nabla.** *Ton wrote:* Almost all procedural textures in Blender use derivatives for calculating normals for texture mapping (with as exception “Blend” and “Magic”). The texture normal, the derivative, is calculated by using four samples in the texture formula:

```
s0= texture(x, y, z) s1= texture(x+nabla, y, z) s2= texture(x, y+nabla, z) s3= texture(x, y, z+nabla) normal[0]= s0-s1 normal[1]= s0-s2 normal[2]= s0-s3
```

Up to now, the “nabla” offset was a constant (0.025) which worked fine in most cases, but doesn't give proper control over the way a texture is sampled, for example to make the effect smoother or sharper. This feature especially is useful in combination with the ColorBand feature.

- **Non-Linear Animation** (NLA) allows the animator to edit motions as a whole, not just the individual keys. Nonlinear animation is not just about editing and manipulating groups of keyframes, but it also allows you to combine, mix, and blend motions to create entirely new animations.
- **Nonuniform Rational B-Splines** (NURBS) is a computer graphics technique for generating and representing curves and surfaces.
- **Normal** (Surface Normal) to a flat surface is a three-dimensional vector which is perpendicular to that surface. A normal to a non-flat surface at a point p on the surface is a vector which is perpendicular to the tangent plane to that surface at p.
- **Normal Mapping** is similar to Bump Mapping, but instead of the image being a greyscale heightmap, the colours define in which direction the normal should be shifted, the 3 colour channels being mapped to the 3 directions X, Y and Z. This allows more detail and control over the effect.

5.1.15 O

- **Orange** is the first Blender open movie project.

- **Oversampling** (OSA), also called **Anti-Aliasing** is the technique of minimizing aliasing when representing a high-resolution signal at a lower resolution. In most cases, anti-aliasing means removing data at too high a frequency to represent. When such data is left in a signal, it causes unpredictable artifacts.

5.1.16 P

- **Phong** shading term is used indiscriminately to describe both an illumination model and an interpolation method in 3D computer graphics. Phong reflection is a local illumination model and can produce a certain degree of realism in three-dimensional objects by combining three elements - diffuse, specular and ambient for each considered point on a surface. It has several assumptions - all lights are points, only surface geometry is considered, only local modelling of diffuse and specular, specular colour is the same as light colour, ambient is a global constant.
- **Point Light** is a light that has a specific location and radiates equally out in all directions. Examples of point lights would be candles or bare lightbulbs. Surfaces close to the point light are brighter than those which are far away. Point lights have attenuation, which controls how quickly the light intensity drops off as you move away from it. Lights with high attenuation are very localized, while lights with low attenuation will spread farther.
- **Polygonization** (*of meta-surfaces*) is the process of approximating the meta-surface via polygons so it can be displayed/rendered in Blender.
- **Purple** runs as a normal Verse client. It implements a node database to mirror the contents of its host. It loads the plug-ins, which reside in libraries, from local disk as DLLs or shared objects depending on the platform.

5.1.17 Q

- **Quaternion** is a representation of 3D rotations with four numbers. It can be interpreted as an extension of complex numbers to 3D. The interpretation of the four numbers is not very intuitive for a human, but the numerical advantage of quaternions is that it is the smallest mathematical representation that does not suffer from the **gimbal lock singularity** problem. This problem occurs for example with **Euler angle** representations, when a small change in the 3D orientation can give rise to a large change in Euler angles.

5.1.18 R

- **Radiosity** is a more accurate but also more process-intensive technique than raytracing, that calculates patterns of light and shadow for rendering graphics images from three-dimensional models. One of the many different tools which can simulate diffuse lighting in Blender.
- **Raytracing** works by tracing the path taken by a ray of light through the scene, and calculating reflection, refraction, or absorption of the ray whenever it intersects an object in the world. More accurate than Scanline, but much slower.
- **Render:** to generate actual viewable images from a 3D model or scene. This may or may not need to happen in real time; for example an interactive game requires real-time rendering, whereas a feature film does not. These situations require very different rendering techniques.
- **Refraction** in geometric optics is the change in direction of a wave due to a change in velocity. It happens when waves travel from a medium with a given refractive index to a medium with another. At the boundary between the media the wave changes direction; its wavelength increases or decreases but frequency remains constant. For example, a light ray will refract as it enters and leaves glass.
- **Relative Vertex Keys (RVK)** are part of a keyframe animation system that operates on vertex level objects. Each (shape) key is stored as a morph target such that several keys may be blended together to achieve complex mesh animation. With RVK you can create facial expressions, speech, and other detailed animated keyframed movements within your mesh-based models.
- **Rig:** the controls that aid in the manipulation of a digital character.
- **Rigging** is the process by which a person creates constraints and relationships between objects that will generate controls to aid in the manipulation of a digital character.

5.1.19 S

- **Scanline** is one row of pixels in the final render. Also the term for one of the methods of rendering which Blender can use. It is much faster than Ray-tracing, but allows fewer effects, such as reflections, refractions, motion blur and focal blur.

- **Seed:** a starting number for generating a random-looking number sequence. Using the same seed will always give you the same sequence. Technically, such a sequence is not truly random, it is only *pseudorandom*.
- **Shader:** an algorithm for computing the appearance of a given material based on the colour, angle and intensity of the light. *Specular* shaders produce a more shiny, mirrorlike finish, while *diffuse* shaders give a duller surface appearance. There are also *toon* shaders, which are deliberately designed to produce an effect more akin to a cartoon drawing, with delineated object borders and less gradation of colours over a surface.
- **Shadows:** simulated lights don't normally cast shadows. And, they also pass through solid objects - so a light inside a closed box would actually illuminate things outside the box as if the box were transparent. The shading on objects is only calculated based on the angle of the surface.
- **Specular Light** refers to the highlights on reflective objects, like diamonds, billiard balls, and eyes. Specular highlights often appear as bright spots on a surface, at a point where the light source hits it directly. Ambient, Diffuse, and Specular are called the three components of a light source. Each one is given a color, which, when added together, create the final color of a light. For most lights, the main overall color of the light is defined by the Diffuse color. Sunlight or lightbulbs would be white, while moonlight would be a darker blue, and a candle would be yellow. You can use the ambient color to adjust the overall color range of the light source; or, you can get a slight tint to shadows by making the diffuse component yellow and the ambient a slight blue. In many lights, the ambient color is left at black, meaning that it won't have any effect. Specular components are often left at white, but you can make them different colors to get interesting effects. Most of the time you can completely ignore the specular and diffuse settings on a light, but just be aware that the way you set the color is by specifically setting the diffuse color. The final color that an object appears to be is a combination of the light hitting it and the color of the surface.
- **Spotlight** is a light with both location and direction. A spotlight sends out a cone of light defined by the spotlight angle, and illuminates only objects within that cone. Spotlights also have attenuation, as well as a parameter that controls whether the spot of light is sharply defined or has smooth edges. These 4 types of lights are listed in order of computational complexity; the more lights you have, the more work the computer has to do. Generally it's a good idea to use

directional lights whenever possible, since they're the cheapest, and use pointlights and spotlights sparingly.

- **Stucci** is one of the classes of blender textures. *Stucci* is not an English word, but is used in Blender as the plural of *stucco*.
- **Subdivision Surface** (*Subsurf*) is the tool which subdivides your model at render-time, without affecting your mesh at design-time. There are two subsurf algorithms in Blender to choose from - *Simple Subdiv*, which doesn't affect the shape of your mesh, and is used to add detail to displacement mapping or render-time radiosity, both of which operate on a per-vertex basis. The other is *Catmull-Clark*, a common subdivision algorithm which smooths out curves, and allows you to make complicated smooth surfaces (e.g. people, plants, etc.) with very few faces. However this algorithm can sometimes (read: often) have strange results with meshes containing triangles or vertices with many edges ("poles"), unless it is correctly handled.
- **Sub Surface Scattering** (SSS) is a mechanism of light transport in which light penetrates the surface of a translucent object, is scattered by interacting with the material, and exits the surface at a different point. All non-metallic materials are translucent to some degree. In particular, materials such as marble, skin, and milk are extremely difficult to simulate realistically without taking subsurface scattering into account.

5.1.20 T

- **Tuhopuu** is an experimental version of Blender that is like a code playground, developers can put their new code in there to be tested and played with by users before it gets put into the official Blender. Tuhopuu is Finnish for "Tree of destruction".
- **Tweening** is short for in-betweening, the process of generating intermediate frames between two images to give the appearance that the first image evolves smoothly into the second image. Tweening is a key process in all types of animation, including computer animation. Sophisticated animation software enables one to identify specific objects in an image and define how they should move and change during the tweening process. Another word for tweening is *interpolation*.

5.1.21 U

- **UV Mapping** (UV) This refers to the process of re-parameterizing a 3d object with dimensions x, y and

z into a 2d plane with coordinates *u* and *v*. Most texturing requires this step because it tells the program HOW to apply a 2d image map onto a 3d object. If all your textures must be 2d and flat, the easiest way to determine what pixel goes where is if your model is flattened and made 2d. It also establishes a relationship between a 2d image and the mesh such that if the mesh deforms, the image map will deform along with it. Think of it as skinning a cat and pinning its hide onto cardboard to facilitate painting it!

5.1.22 V

- **Verse** is a network protocol that lets multiple applications act together as one large application by sharing data over a network. If one application makes a change to shared data, the change is distributed instantly to all the other interested clients.

5.1.23 W

- **WC** is Weekend Challenge.
- **WIP** is Work In Progress.

5.1.24 X

5.1.25 Y

- **Yet Another Free Raytracer** (YafRay) is an open source ray tracing program that uses an XML scene description language. It has been integrated into, and is often used to render scenes made in, Blender.

5.1.26 Z

5.2 Frequently Asked Questions

The following questions cover general information about *Blender 3D: Noob to Pro* and the software it is designed to teach, *Blender*.

What is Blender?

Blender is a full-blown 3D content creation suite, usable for 3D editing, animation, texture mapping, compositing and rendering; as well as 2D video and audio editing/sequencing.

It runs on:

- Windows — XP, Vista, and 7, 32- and 64-bit
- Mac OS X — PowerPC and Intel, 32- and 64-bit

- Linux — Intel x86 and AMD, 32- and 64-bit
- BSD
- Solaris
- Irix.

It is free, open-source software, available from blender.org.

Blender **2.73a** is the latest stable release as of January 2015. Beginners are urged to learn using stable 2.5x versions, as there has been a significant UI overhaul since 2.49 and future tutorials will likely be written or recorded with the new interface. However, keep in mind that many existing tutorials were written for 2.49. If you're having trouble finding a button or tool that has moved, try pressing Spacebar to search for Blender functions.

For more about Blender's history, functionality, and media productions which have utilized Blender, see the [Blender entry in Wikipedia](#).

What is this book? And what isn't it?

Noob to Pro is a collection of tutorials. Following these tutorials you can go from ignorant new Blender user (noob) to proficient Blender power user (pro). Though the tutorials are not strictly linked, we recommend going through them in order. Earlier tutorials introduce skills and information you'll be expected to know in later sections.

Noob to Pro isn't a compendium or reference manual. Use it with other resources, including those in Blender's Help menu. *Noob to Pro* provides links to outside resources. We're part of the Blender community!

How accurate is the information in this book?

Blender is still under development! Its appearance, tools, and capabilities change from time to time. For instance, Blender 2.5x alters both the graphical user interface and the Python scripting interface and may also change some keyboard shortcuts.

Most of the tutorials should work with any recent version of Blender. *Blender 3D: Noob to Pro* readers are encouraged to edit the WikiBook in order help to keep it up to date.

I have a question, where can I get help?

Search or post questions at [BlenderArtists](#) a large, easy-to-use, English-language forum. [BlenderWiki](#) contains a large database of documentation. Find similar forums in other languages at [blender.org's community page](#). The discussion pages there are also useful.

If your question turns out to be very common, please take a moment to see if it can be added to the book! It is best if it can be smoothly incorporated with an existing page.

5.3 Tutorial Links List

This page includes a categorized list of Blender tutorials written or spoken in English. This is by no means a comprehensive list of tutorials, so adding links is encouraged. For tutorials that are not in English, please refer to the [About page](#).

If you would like to learn about editing pages, you can find information in the [Introduction to Wikipedia](#).

5.3.1 General Blender Information

- [Blender Documentation](#) - Includes information on Blender functionality, developing for Blender, and creating Addons.
- [To Those Learning 3D](#) - Recommendations for people first diving into Blender.

5.3.2 Blender Tutorial Websites

This section includes links to websites that host multiple tutorials or provide links to other Blender tutorials. This section does not contain links to individual tutorials.

- [BlenderGuru](#) - Run by Andrew Price, BlenderGuru includes free tutorials as well as full training courses.
- [Blender Cookie](#) - Part of the CG Cookie Network. Includes many free tutorials. Other tutorials and full courses are available to members with a subscription.
- [Dark Scarab Tutorials](#) - Many free tutorials covering various topics in Blender.
- [BlendTuts](#) - Free video tutorials on a wide array of topics.
- [BlenderDiplom](#) - Free tutorials mostly focused on visual effects, but includes other topics as well.
- [Blender Tutor](#) - Includes a series for Blender beginners as well as tutorials for more advanced users
- [BlenderArt Magazine](#) - Blender magazine that includes tutorials in each issue.
- [Blenderpedia](#) - Free video tutorials, mostly about creating scenes and environments.
- [3D Total Blender Tutorials](#) - Includes many tutorials about various topics in Blender.

- [BlenderCourse](#) - 120 page Blender E-Book created for Blender 2.63
- [Tutorials For Blender3d](#) - Many Blender Game Engine tutorials for Blender 2.6
- [Blender Tutorials Vimeo Group](#) - This group collects video tutorials posted on Vimeo.
- [Blender Underground](#) - Many downloadable videos that cover the basics of Blender.
- [Super3boy's Blender Tutorials](#) - Video tutorials dealing with the basics of different aspects of Blender.
- [Greybeard's Blender Video Tutorials](#) - Downloadable video tutorials that go over many topics.

These sites include links to Blender tutorials on other sites

- [BlenderNation Tutorials](#) - List of tutorials that have been posted on BlenderNation
- [Blender3d Club](#) - Includes links to various Blender tutorials
- [SurfnLearn.com : Blender 3D Tutorials](#) - Links to various Blender Tutorials
- [Learningblender.com](#) - Categorized links to various Blender Tutorials

These sites include tutorials that use Blender versions prior 2.5.

- [Dave Jarvis' Blender Tutorials](#) - Free tutorials using Blender 2.4x
- [Kator Legaz](#) - List at bottom includes a couple Blender tutorials
- [BlenderMasters](#) - Links to tutorials and some embedded YouTube tutorials. All 2.4x and older.
- [Selleri](#) - Handful of tutorials on a very old version of Blender.
- [Cogfilms Tutorials](#) - PDF downloads of tutorials in 2.4x
- [Pavaler.se](#) - A couple of tutorials on modeling and animation.

5.3.3 Blender Interface

- [Blender Basics: 2 Minute Tutorials](#) - The Interface
- [Blender Interface Theme Repository](#)
- [Basic Editing](#)

- [Blender Hotkeys](#)
- [Changing Views](#)
- [First Impression](#)
- [Getting Started](#)

5.3.4 Modeling

Mesh Modeling

- [Video - DupliVerts Tutorial](#)
- [How to solve Blender's smoothing problems: discussion about smoothing strategies: Smooth/Solid, Auto Smooth, EdgeSplit modifier...](#)
- [Heightmaps](#)
- [Extrusion Controlled by IPO](#)
- [Basic objects](#)
- [Modeling Techniques and Strategies](#)
- [Easily Remove Orphan Edges and Vertices](#)
- [Non-destructive bevel effect](#)

Nurbs and Subsurface Modeling

- [Subsurf modeling 2](#)
- [Metaballs](#)
- [Abstract SubSurf modeling I](#)

Modeling Specific Objects

- [Wooden Crate](#)
- [Double-Helix](#)
- [Turtle](#)
- [Leaf Shader](#)
- [Sears-Roebuck Dairy Barn](#)
- [Golf Ball](#)
- [Celtic Knot](#)
- [Quick and Easy Tree](#)
- [Ice Cube \(*pdf tutorial*\)](#)
- [Water I \(*pdf tutorial*\)](#)
- [Water II \(*pdf tutorial*\)](#)
- [Dice](#)
- [Die with round corners](#)

- Logo
- Cutting through steel
- Candle
- Landscape Cartoon (*pdf tutorial*)
- Text
- Hair Tutorial
- Wikipedia: Human Body Proportions

5.3.5 Texturing

- Video - Basic Materials
- Making a Lightsaber with Halos
- The Unofficial Texturing Tutorial
- Material Indice Tutorial
- Transparency and Refraction
- Reflections
- Texturing landscapes according to slopes
- Using Texture Stencils
- Textures with Alpha
- Textures with Bumpmapping

UV Mapping

- 2 Minute Tutorial: Blender Basics - UV Mapping (Project from View)
- Video - UV Mapping
- UV Mapping & Texturing
- Complex material for a sword

Animated Textures

- Video - Changing Colors Of Objects Mid Animation
- animating masks for simulating ice freeze
- Animated Procedural Textures (*pdf tutorial*)

Texture Nodes

- Video - Blender Nodes
- Vector Blur usage with Blender

2D Texture Painting Techniques

These tutorials do not directly involve Blender, however they cover useful 2D knowledge for advanced Blender users.

- Steven Stahlberg's Tutorials * exposed
- Fixing Lighting Irregularities in Self-Tiling Maps
- Texturing with Gimp

5.3.6 Lighting, Shadows and Rendering

- 2 Minute Tutorials: Blender Basics - An Outline of Lighting and Rendering in Cycles
- Basic 3 Point Lighting
- Radiosity I
- Radiosity II
- Ramp Shaders
- Simulating Radiosity
- Ambient Occlusion
- Soft Shadows and Area Lights
- Soft Lights (*pdf tutorial*)
- Blender's Mist

5.3.7 Compositing

- Matching Real Lighting

5.3.8 Animation

- Video - Animation Basics
- Simple Animations
- Effects
- Walking Blues
- Non Linear Action Editor - NLA I
- Non Linear Action Editor - NLA II

Armatures and IK

- Video - Basic Armatures
- Dancing Flor * exposed
- Animation Workshop II * exposed
- Animation recode project
- Action Constraints tutorial made easy
- Driven Hand Rig
- Constrained Mechanics
- Ikas Blender 3D - Introduction

5.3.9 Particles

- Video - Particle Basics
- Video - Grass with Particles
- Video - Explosions with Particles and the Explode Modifier
- Particle Interaction
- Static Particles
- Making a Rain Effect
- Setting a prey-predator relationship using Boids particles
- Particle and field basics

5.3.10 Fluid Simulation

- Video - Fluid Simulation
- Fluid Simulation Basics

5.3.11 Game Engine

- Video - Blender Game Engine Basics
- Video - Making Dominoes
- Game Engine Development
- New fully integrated game engine?
- #GameBlender
- Walkthrough Tutorial
- Mouse Look 1st Person. Options include capping, invert mouse pitch and adjust mouse sensitivity.
- Viewports. Creating a split screen.
- Python game functions. Sample Code Included. Updated to 2.48

- Car Setup. Use Bullet Physics vehicle wrapper.
- Python scripting for the Game Engine at cgmasters.net.
- Python scripting series of tutorials

5.3.12 Python and Plugins

- Povanim Export Script
- AI Path Importer
- 3D-No Plugins; Put your 3D Blender space on web!
- Randomizer Script Scroll down the page
- Different Useful Scripts
- Lsystem tree maker
- Python API Introduction
- Python API, Making a Square Mesh
- Python API, Iterations
- Python API, Automating Vertex Creation
- Python API, Automating Face Creation
- Python API, Making Potatoid
- Python Script, To build an Empty for EnvMap
- Python Script, Bezier Curves Import
- Python Script, Paths import
- Python Script, Importing Adobe Illustrator Format
- Python Script, Mesh Explosion
- Python Script, Level Of Detail
- Python Script, Wire Shadows and extrusions
- Python Script, Changing the active camera instantly
- Using the Superficial Scattering Script
- City Block Generator
- Blender Camera Calibration with Live Camera
- Mesh shaker and tutorial
- Batch STL
- Vertices to a Curve Converter
- Zutils, Z-Buffer Utilities

5.3.13 Related Programs

- YafRay
- SPE - Python IDE for Blender
- Wings 3D Subdivision Modeler
- Verse Gimp-Blender Plugin
- Kerkythea Renderer
- Equinox 3D
- Povray export
- Adobe Illustrator Paths import
- Inkscape SVG import
- Voodoo Camera Tracker
- Lionhead The Movies

Distributed Computing

- BURP - Big and Ugly Rendering Project
- Global Rendering-Farm

5.3.14 Blender WikiBooks

- Blender 3D: Blending Into Python
- Blender 3D: HotKeys
- Blender 3D: Import and Render a SolidWorks Model
- Blender 3D: MemoBook * exposed
- Blender 3D: Noob to Pro

5.3.15 FAQ

- Blender FAQ (Generated on September 24, 2001)

5.3.16 Miscellaneous

These are must reads for CG artists.

- How to succeed in Animation
- Classical Film and Video Knowledge Base
- Quick Tips in Design&GFX

Open Movies

- NaNo - Blender Internet Virtual Movie Studio

IRC

- irc://irc.freenode.net/blender
- irc://irc.freenode.net/blenderchat
- irc://irc.freenode.net/blenderclasses
- irc://irc.freenode.net/blenderqa
- irc://irc.freenode.net/blendercoders
- irc://irc.freenode.net/blenderwiki
- irc://irc.freenode.net/gameblenderdev
- irc://irc.freenode.net/gameblender
- irc://irc.freenode.net/verse

Tests

- Blender Benchmarks

5.3.17 Other Lists

Please read more about on [talk](#) page.

- [LANGUAGES \(czech\)](#)

5.3.18 About

This links list is language filtered and extended version of personal collection originally provided by [IamInnocent](#). So if you looking for tutorials in other languages check this link at www.elYsiun.com.

- German Blenders should have a look at http://de.wikibooks.org/wiki/Blender_3D:_Tutorial_Linkliste

If you want to add Blender tutorial link in some other language you can add it temporary on [talk](#) page. We will later made such WikiBooks in other languages too.

Thank you all who contributed to this nice and useful links collection! Feel free to add your name or link if you think you need to be mentioned here.

5.4 Hotkeys

Blender's user interface is very flexible and often provides more than one way to do something. Many operations that can be done by clicking on buttons and menus can also be done with keyboard shortcuts called "hotkeys". The hotkey is usually much faster than the corresponding mouse action, so it's a good idea to learn them as soon as possible.

An entire WikiBook (titled *Blender 3D: HotKeys*) is devoted to Blender hotkeys. We encourage you to refer to it as often as necessary.

5.5 Output Formats

Blender can render still images and video to many different file formats. Here is an alphabetical list:

- **AVI Codec** - AVI codec compression. Available codecs are operating system dependent. When an AVI codec is initially chosen, the codec dialog is automatically launched. The codec can be changed directly using the Set Codec button in AVI Codec settings.
- **AVI Jpeg** - AVI but with Jpeg compression. Lossy, smaller files but not as small as you can get with a Codec compression algorithm. Jpeg compression is also the one used in the DV format used in many digital camcorders.
- **AVI Raw** - Audio-Video Interlaced (AVI) uncompressed frames.
- **BMP** - Bit-Mapped Paint is an early lossless format.
- **Cineon** - format produced by a Kodak Cineon camera and used in high-end graphics software and directed toward digital film.
- **DPX** - Digital Moving-Picture eXchange format; an open professional format (close to Cineon) with meta-information about the picture; 16-bit uncompressed bitmap (huge file size). Used in preservation.
- **FFMPEG** - Fast Forward Moving Pictures Expert Group is a collection of free and open source software libraries that record, convert and stream digital audio and video in numerous formats. You must have the proper codec installed on your computer for Blender to call it and use it to compress the video stream through FFMPEG, but there are preset formats to choose from, such as DV, SVCD, and DVD.
- **Frameserver** - Blender puts out frames upon request as part of a render farm. The port number is specified in the OpenGL User Preferences panel.
- **HamX** - Blender's own self-developed 8 bits RLE (Run Length Encoded bitmap) format; it creates extremely compact files that can be displayed quickly. To be used only for previsualization of animations (Play button).
- **Iris** - the standard Silicon Graphics Inc (SGI) format used on Unix OS machines.
- **JPEG** - Joint Photographic Experts Group (name of the consortium which defined it), an open format that supports very good compression with little loss of quality. Only saves RGB value. Re-saving images results in more and more compression and loss of quality.
- **JPEG 2000**
- **MultiLayer** - an OpenEXR format that supports storing multiple layers of images together in one file. Each layer stores a renderpass, such as shadow, specularity, color, etc. You can specify the encoding used to save the MultiLayer file using the codec selector (ZIP (lossless) is shown and used by default).
- **OpenEXR** - an open and non-proprietary extended and highly dynamic range (HDR) image format, saving both Alpha and Z-depth buffer information.
- **PNG** - Portable Network Graphics, a lossless image format often used in web design.
- **QuickTime** - A proprietary multimedia framework for video and images.
- **Radiance HDR**
- **Targa**
- **Targa Raw**
- **TIFF** - Tagged Image File Format created by Aldus for desktop publishing.

If the selection seems daunting, here are pointers:

If you need transparency in your images (i.e., an alpha channel that allows any part of the rendered image to have varying degrees of opacity), or you will be compositing, you generally want to use:

- Multilayer
- PNG
- OpenEXR
- Targa

Of the above four formats, you will find that PNG and Targa images can be opened and edited by most any bitmap editing application, such as GIMP or Photoshop. However, OpenEXR and Multilayer offer more options within Blender, and may produce smaller, lossless files.

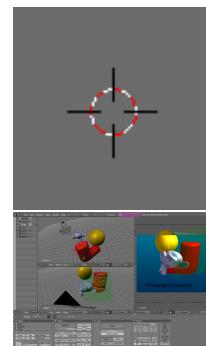
5.6 Image Portfolio

5.6.1 Summary

This is the *Blender 3D: Noob to Pro* image portfolio page, where all the categories of the images used in the book are here in one place for your reference.

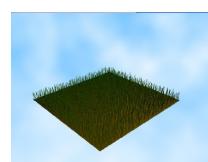
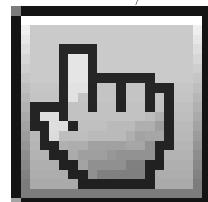
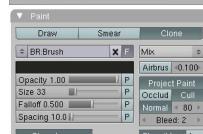
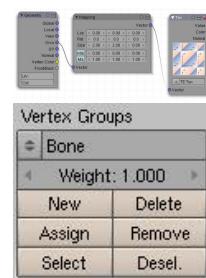
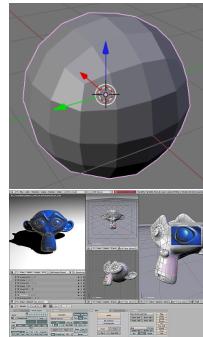
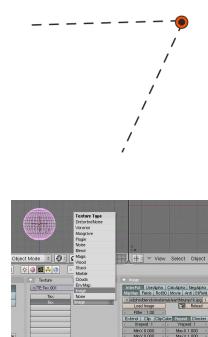
5.6.2 Categories

- Blender 3D
- Blender 3D icons
- Blender 3D screenshots
- Non Blender 3D screenshots
- Blender 3D splash screens
- Blender 3D viewports
- Blender 3D suzanne primitive

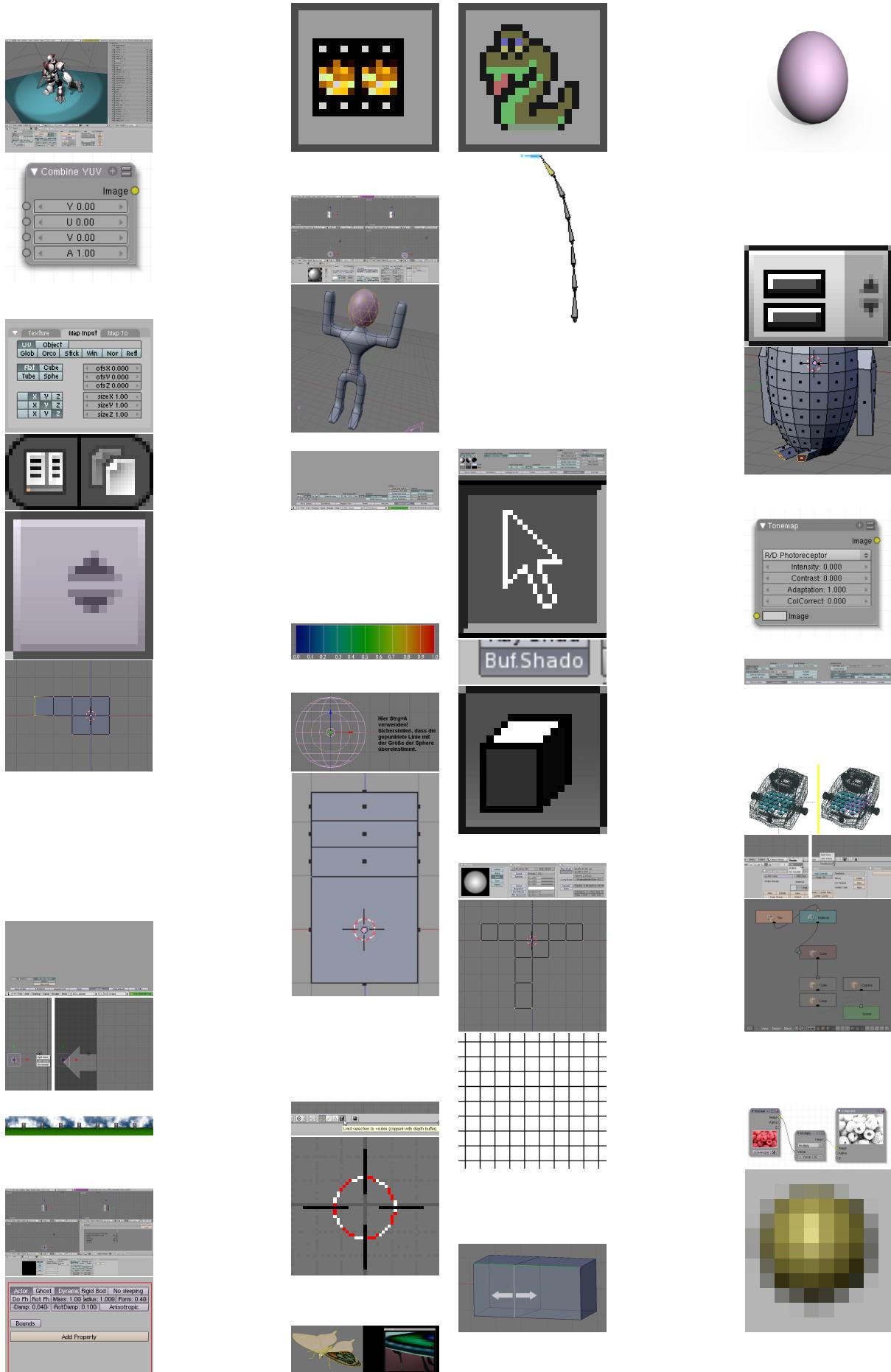


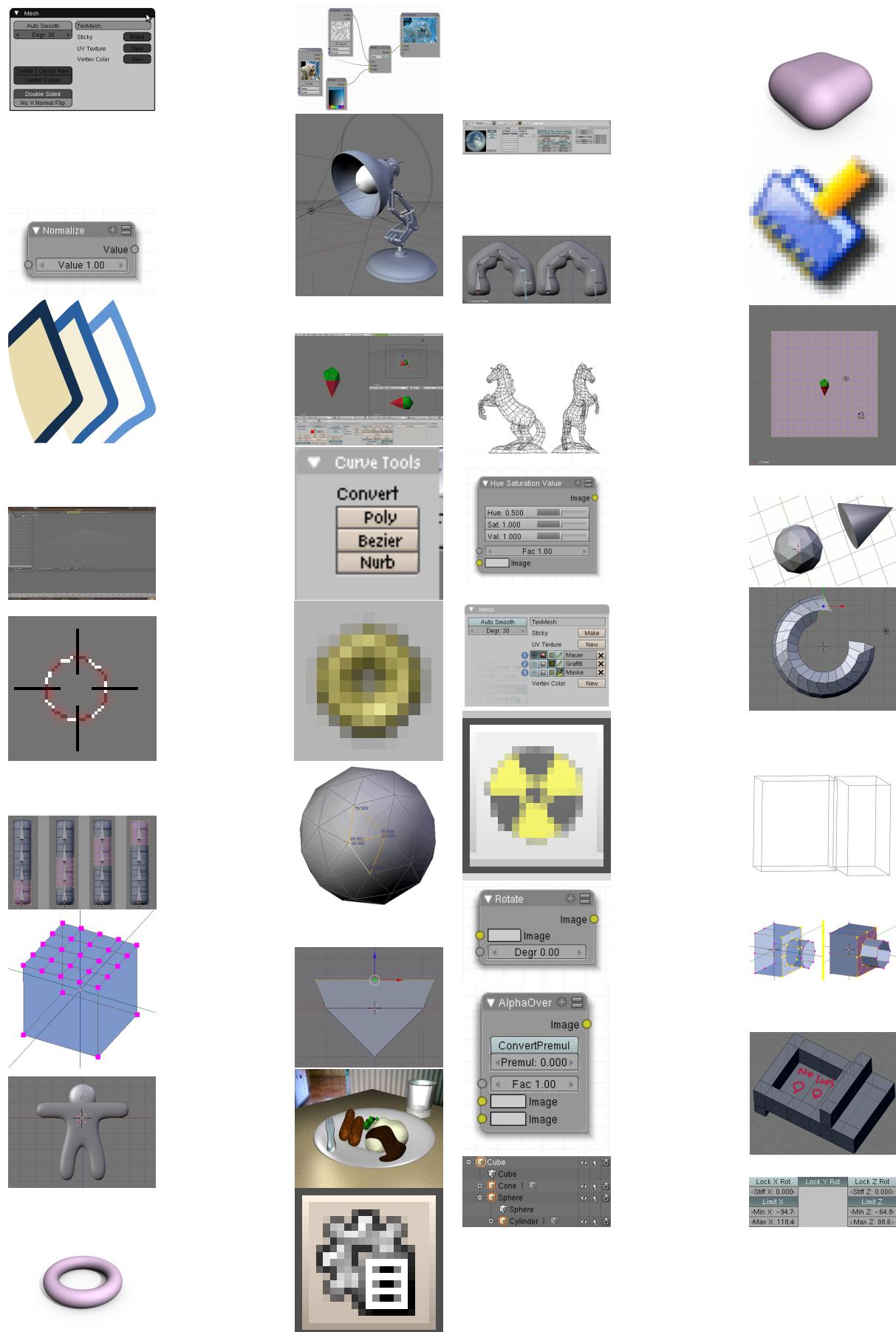
5.6.3 A sampling of images from the above categories

Heres a random sample of 200 images from the over 1,800 images in the above categories

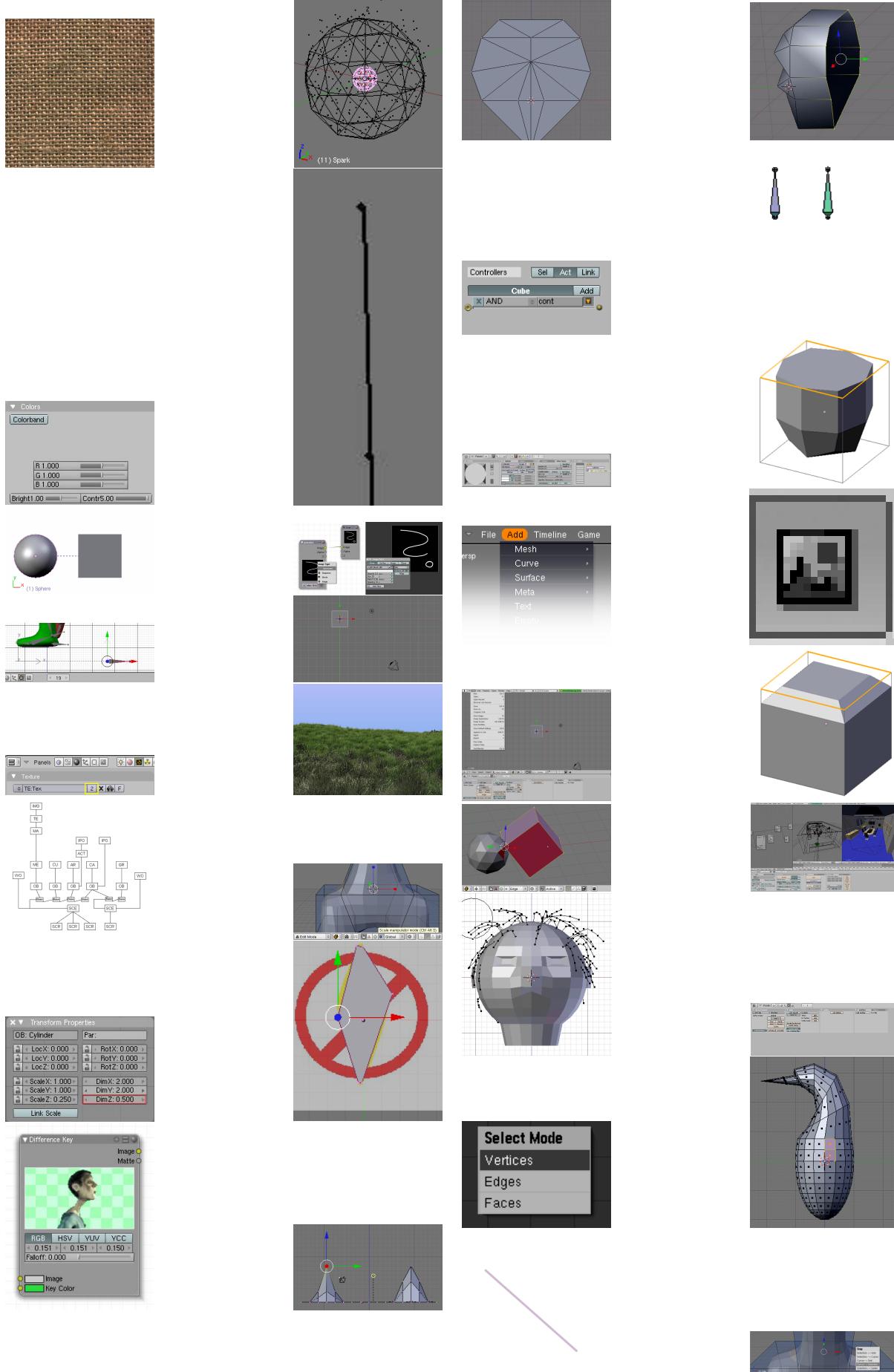


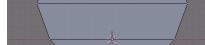
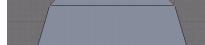
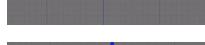
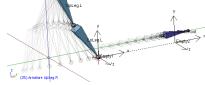
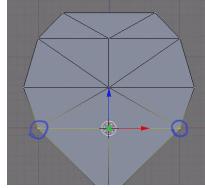
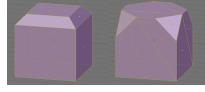
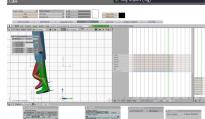
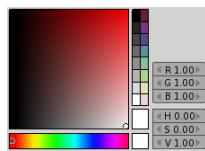
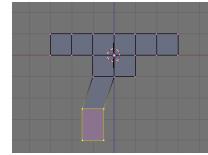
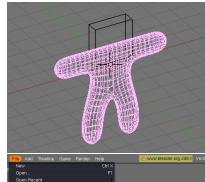
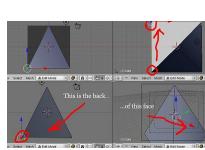












- **perspective view**: a 2D view which represents a scene's parallel edges with edges that converge toward a vanishing point
- **pose** (verb): to position objects and their parts in a 3D scene
- **render** (verb): to generate a 2D view from a 3D model of a scene, by means of a computer
- **rig** (noun): to describe how different parts of an object are capable of moving relative to one another
- **texture** (verb): to describe how a 3D object interacts with light (scattering, bending, etc.)

Blender User Interface Terminology

- **active window**: the window currently responding to keystrokes
- **blender unit**: the height and width of each grid square in a 3D View window
- **header**: a row of controls along the top or bottom of a window
- **panel**: a rectangular area in a Buttons window which can be rearranged relative to other panels
- **window** (noun): a rectangular area of Blender's user-interface which can be split or resized
- **window type** (noun): one of the sixteen varieties of windows in Blender's user interface:
 - **Scripts Browser** - This window is used to initiate any [Python](#) scripts saved in Blender to add functionality.
 - **File Browser** - Browses the files on your computer in Blender.
 - **Image Browser** - Browses the files on your computer in Blender with visuals of images displayed.
 - **Node Editor** - The enhanced material editing system in Blender in which you can customize the input/output of materials or scenes and composite them together.
 - **Buttons Window** - Displays the Blender scene and object editing options.
 - **Outlines** - Breaks down the entire collection of objects and their properties in an organized list.
 - **User Preferences** - Allows you to change Blender's controls and preferences.
 - **Text Editor** - Simple text editor for any notes/Python code.

- **Audio Window** - Allows you to upload, enhance, and save audio files into the Blender library.
- **Timeline** - Shows the animation's beginning, ending, and current frame, as well as a timeline with marks indicating all the key positions of an object.
- **Video Sequence Editor** - Feature for the composition of sound, scenes, videos and other cinematographic elements.
- **UV/Image Editor** - Allows you to add and alter image files in Blender and also apply these images onto the faces of meshes in an editable fashion as a material.
- **NLA Editor** - Allows you to change the key positions of all actions/IPOs.
- **Action Editor** - Allows you to alter the key positions of an armature and its collection of bones.
- **IPO Curve Editor** - Allows you to insert saved key points of the positions, transformations, and settings of objects of the course of a series of frames.
- **3D View** - Window through which you work with, add, and place objects in a 3D world.

Blender Modeling Terminology

- **Quad *n*** - A face with four sides.
- **Mesh *n*** - A collection of faces, edges, and vertices which can be modeled and manipulated in Edit Mode.
- **Vertex (*pl. vertices*) *n*** - A 3-dimensional coordinate, which in groups comprise a polygon. With default settings, it is represented in Blender by a purple dot when unselected, or a yellow dot when selected.
- **Edge *n*** - A wire-like line representing the boundary of 2 adjacent vertices.
- **Face *n*** - A planar connection of edges representing a boundary, field, or solid surface. Identified individually in Edit Mode as a black box in the middle of the corresponding face.
- **Icosphere *n*** - A sphere composed of *n* triangular faces composed in a manner to give the best "smoothness" for the lowest memory consumption.
- **UVSphere *n*** - A sphere composed of *n* square faces arranged in rings to allow the smoothest application of real-time movement of vertices/faces and application of images onto the spectrum of faces.
- **Lamps** - Category of 5 light-emitting objects:

- Lamp - Light is emitted in the scene with a uniform spherical output which fades with distance, decreasing eventually to 0. The distance can be specified in its options.
- Sun - Light is emitted in the scene with a uniform spherical output and remains uniform regardless of the distance from it.
- Spot - Light is emitted in the scene in a pre-determined conical area with light properties similar to lamp and also initially calculates the shadow (lack of light) present behind an object (Known as ray shadow).
- Hemi - Sun-like light emitted in a 180 degree arc. Works just like a Sun lamp, but with less functionality. Best used for quick, broad illumination.
- Area - Light which works similar to a Spot lamp but shape of the cone is definable.

- **Ray shadows n.** - Method of generating shadows cast by light by calculating the angle and faces of the interfering object to create a smooth, sharp and complete shadow. Memory-intensive, best quality.

- **Buffer Shadows n.** - Method of generating shadows cast by light by calculating the angle of a mesh's edge and creating a set amount of shadow "squares" on the shadow-receiving objects which are calculated from a set buffer-ratio. Memory-friendly, while possibly lower quality.

- **Camera n.** - An object which acts as the viewpoint (when active) for the scene when the scene is to be rendered.

- **Armature n.** - An object which contains 'bones'. Bones are sub-objects which are connected with each other in a chain, each bone-link is parented to the bone before it. The purpose is to parent any objects to the bone, and also you may parent a mesh to the entire armature and have selected vertices of the mesh parented to bone(s) percentage-wise (0% means no effect, 50% means vertices copy half the transformation value, 100% means vertices transformed as full child of bone.)

- **Empty n.** - An object with no mesh or relating object data. It is useful for using in object input fields where the object needs no specific properties e.g. parenting it to an object for organisational purposes.

5.8 Materials Directory: Every Material Known To Man

5.8.1 “Every Single Material Known To Man” Project

Want to share your material settings with the world? This is the place! This page is going to be turning from a page to a chapter, but to do that we need your support! If you have an idea for a material, even if you don't know how to make it, just put it in the list to help! Someone out there knows how to make it, and if they come across this page with it listed, they might be willing to share.

Etiquette

- Explain the basics of every material (such as Color, Spec, and shading, but not necessarily specifics such as Emit or Trans), even if it is with standard “Add New” properties.
- IF you provide a picture, explain in the method all details to get us to make ours look the exact same way, do not omit any detail (including anything done in out tabs, or in other categories like Rendering. Also, say if you use something like RayMir or if you have additional lighting or objects to enhance the look!).
- Make sure you explain the “Tab” and the “Category” for a material alterations, because not everyone might know what you mean. Be as formal as possible, categorize and organize, and try using the blender-used words instead of your own lingo, if possible.

Example:

- Bad explanation:*Set the material's texture's normal to 0.4*
- Good explanation:*Under the “Map To” Tab, Select the “Nor” Slider and set it to 0.4*
- Best explanation:
 - **Map To:**
 - **Output-Nor**
 - **NorSlider-0.4**
- Do NOT replace or delete other people's method on your own. If you believe you have a better way, simply label their explanation “Method A” and create another sub-headline with your method as “Method B” (and so on). If you believe a method is incorrect, insufficiently detailed or plagiarism, appeal to the original author (if able) or on the Discussion page and get support for a removal.

- Do NOT say your material is better than anyone else-- everyone has a right to their own opinion and preferences --you may say your method is more “detailed” or “memory-efficient” or “adaptable”, but DO NOT SLANDER other people’s methods.
- Add or link the texture image file (png), if there is.
- Add or link a render example image of the material (just a sphere or a cube or a monkey) with a background for contrast.

5.8.2 Metals and Minerals

- Alloy
- Asphalt
- Brass
- Brick
- Bronze
- Cast Iron
- Ceramic
- Chrome
- Glass
- Gold
- Iron
- Concrete
- Copper
- Hard Plastic
- Linoleum
- Marble
- Reflectives
- Modeling Clay
- Rubber
- Rust
- Steel
- Tile
- Trampoline material
- Realistic Silver
- Jewels

5.8.3 Cloth and Fabrics

- Cotton
- Silk
- Leather
- Spandex
- Wool
- Carpet
- Denim
- Mothball
- Lace
- Dusted Latex
- Glossy Latex
- Felt
- Kevlar
- Snake Skin
- Velvet

5.8.4 Particle-Based

- Fire
- Smoke
- Fog
- Fur
- Rain
- Mist
- Grass (Patch)

5.8.5 Earth and Home

- Moss
- Water
- Soil
- Grass(Strand)
- Woods
- Textured Ceiling
- Textured Wall
- Paint (Matte)
- Paint (Gloss)
- Sand

5.8.6 SSS-Used

- Crayon
- Wax
- Fruit Skin
- Milk
- Juice

5.8.7 Other

Lunar Surface Bubbles

5.8.8 Abstract

- Light Effects
- Misty Globe

5.8.9 Human

- African Skin
- Latino Skin
- South Asian Skin
- Caucasian Skin
- Head Hair
- Other Hair
- Teeth
- Bone
- Blood
- General Organ Skin

5.8.10 Elemental

- Selenium
- Mercury

5.8.11 External material libraries

Additional materials can be found at:

- Blender Open Material Repository
- Copper, Brass, Platinum, Silver
- Fire, Snow, Carbon Fibre
- Glass, Chrome, Gold, Brushed Alloy

5.9 Sources of free 3D models

Here are some sources of free 3d models from across the web. Take them apart, put them back together, understand what makes them tick, to improve your own modeling.

5.9.1 Blender-Specific

These sites offer .blend files for download.

- The Official Blender Model Repository
- Blender for Architecture
- <http://opengameart.org> — more Blender models
- http://katorlegaz.com/3d_models/index.php -- they have furniture, arcade video games, joysticks, toys, vehicles, road and sidewalk items, and simple human models
- Yorik's site — some Blender files for his architectural projects are available for download.

Sites Requiring Registration

- <http://www.blendswap.com/> — specifically for swapping Blender models.

5.9.2 Non-Blender-Specific

These sites offer models in other formats.

- <http://archive3d.net> - 15,000+ free models from archive 3d

This seems to have some licensed models and might be legal trouble see Discussion at <http://blenderartists.org/forum/archive/index.php/t-237569.html>

- <http://sketchup.google.com/3dwarehouse/> - Google's 3d warehouse repository; take the collada version, when available (the Google Earth downloads are actually .zip files with a collada inside them, so just rename and unzip)

Note: some of the 3d warehouse models can't be understood by blender due to a zillion repeating elements, others will show up invisible (one sided, or inside out), be gigantic in size, or all parented to some crazy spot out in space. Fixing these to be usable will push your blender skills to new levels in some cases.

Tips: I've tried importing a few of these 3D Warehouse models, and so far I've only noticed a couple of issues:

- The model imports into a new Blender scene, called something like “SketchupScene” or “SketchupScene001”. You may also find additional scenes created with nothing in them, that you can delete. If you try rendering, you may not see anything, because the default camera is only linked into the original default scene named “Scene”. Just select the camera and hit **CTRL+LKEY** to link it into the new scene.
 - You may notice parts of the model appearing and disappearing as you try viewing it from different angles. This is because it is scaled very large (several hundred Blender Units on a side). Try scaling it down so it's no more than a few BU on a side.
 - There will usually be a lot of doubles, so remove them with **WKEY->remove doubles**.
 - There may be many separate objects, so select what you want to combine and hit **CTRL+J** to join them all into one object.
- <https://www.stlfinder.com> - The free 3d model search engine
- <https://www.yobi3d.com> - 3D model search engine with previews in 3D
- <http://free3dbase.com/> - furniture models without registering
- <http://artist-3d.com/> - wide variety of free models
- <http://nasa3d.arc.nasa.gov/> - all kinds of free models from the space program
- <http://www.3dm3.com/modelsbank/> - around 300 free models
- <http://www.amazing3d.com/free/free.html> - free section on amazing 3d about 75 models of all kinds of things.
- http://www.corporatemedianews.com/2002/03_mar/features/3dmodellibrary.htm - free models from corporate media news; a very interesting collection from klingon space ships to footballs.
- <http://3dplants.0catch.com/download.html> - about 70 free 3d plant models
- <http://www.3delicious.net> - 3d delicious, lots of models, but way too many popover ads! (click the picture, then click the bigger preview in the next page to download)
- <http://www.top3dmodels.com/products/3d-models/free> - 20 free models
- <http://www.top3d.net/free-3d-models/> - about 85 models, including washing machines, tires, and jewelry
- <http://www.oyonale.com/modeles.php?lang=en&format=OBJ> - about 17 free models form guns, to rubber duckies
- <http://dmi3d.comeze.com/> - a few hundred free models of cars, race cars, trucks, tanks, backhoes, etc.
- <http://www.scifi3d.com/list.asp?intGenreID=14&intCatID=36> - free models of scfi empires like, Aliens, Babylon5, Blade Runner, BS Galactica, Original SciFi, Other SciFi, Star Trek, Star Wars
- resources.blogscopia.com - models in Blender and other formats.
- <http://www.hobbycncart.com/> - Free relief models (all modells are free) in .STL .OBJ and depth map (.BMP) formats.
- <http://www.craftsmanspace.com/free-3d-models> - Free 3D models (Creative common licenses) in .STEP, .IGES, .STL .OBJ and .BLEND formats.
- <http://tf3dm.com> — models in various common formats, including some in .blend; can be downloaded without registration (subject to slight delay). However, most seem to be under “personal use only” or “non-commercial use” licences.

Sites Using .RAR Format

Separated out because they may not be decodable with Free Software.

- <http://www.3dmodelfree.com/> - 3d model free, lots of variety here
- <http://dlegend.com/html/free-3dmodels.html> - design company with a few huundred free 3d models
- <http://3ddb.blogspot.com/search/label/Models> - lots of free luxury style models in sets (has a giant picture at the top, just scroll down).

Sites Requiring Registration

- http://www.turbosquid.com/Search/Index.cfm?keyword=&max_price=0 - 13,000+ free models at 3d stock site turbosquid (registration required for download)
- [http://www.3dxtras.com/3dxtras-free-3d-models-list.asp?catid=-\\$-1](http://www.3dxtras.com/3dxtras-free-3d-models-list.asp?catid=-$-1) - 3d extras, lots of variety, but too many popover ads.
- <http://dd-freebies.blogspot.com/search/label/3D%20models> - a few baubles and ornaments
- http://www.exchange3d.com/free-3d-models/cat_35.html?noredirect=1&act=viewCat&catId=35 - about 100 free models, some even made in blender!

- http://www.wirecase.com/Gallery-Free-3D-Models_s-2_v-20-1_f-fc-0-321_f-fpx-_f-fpn-_f-oid-.html - 25 sets of models from airplanes to bottles
- <http://www.sharecg.com/b/5/3D-Models?PSID=e3aa02d1ef780c432e9f9d7e95409369> - wide assortment of free models
- <http://www.cg-files.com/3dmodels.html> - from tires to fans (lots of popover ads)
- <http://www.3dcontentcentral.com/default.aspx> - Free 3D Models of Parts & Assemblies like brackets, resistors, motors, shafts, valves.

5.10 All Blueprints Links

This page contains links to 2D images and drawings that can be useful for constructing models.

- <http://www.shipschematics.net> - Starship Schematics Database
- http://www.suurland.com/blueprints_archive.php
- <http://www.luft46.com/profiles/profiles.html> - 3 views
- <http://www.aviastar.org/index2.html> - 3 views
- <http://www.rcgroups.com/forums/showthread.php?t=557457> - 3 views on a lot lot lot of airplanes
- http://www.google.com/intl/en/sketchup/3dwh/pdfs/modeling_a_city.pdf - Google's guide on how to model a city
- <http://www.boatdesign.net/boat-plans-archive/index.htm> boats
- <http://www.mgussin.freeuk.com/00Plans.htm> old cars
- http://www.airwar.ru/other/draw_1w.html War Planes
- <http://jpracing.racerplanet.com/> Cars, Trucks, Buses
- <http://www.swaqvalley.com/blueprints.php?section=samples> sports cars
- <http://www.newhomesource.com> — site run by builders, but low-res plans are viewable online
- <http://drawingdatabase.com/> Vehicles: cars, military, aircraft

5.10.1 Registration Required

- <http://www.the-blueprints.com/> — requires registration for full-quality downloads
- <http://smcars.net/>

5.10.2 Payment Required

- <http://www.monsterhouseplans.com/> - source of over 23,000 floor plans
- <http://www.eplans.com/> - floor plans, but if you navigate around a bit to find em.
- <http://www.emporis.com/application/> - heights and details of buildings; photos available for purchase
- <http://skyscraperpage.com/> - building heights, diagrams, but downloads and screenshots are prohibited
- <http://humster3d.com/category/blueprints/> - cars, trucks, motorcycles, buses

5.10.3 Shouldn't Really Be Here?

- http://www.archiplanet.org/wiki/Main_Page - building encyclopedia, can't find any actual plans

5.11 Materials, Textures, Photos

Here are some sources of materials, textures and photos from across the Web.

Materials

- Large Free Repository of Materials

Textures

- Textures Library
- Mayang's Free Textures Hi-Res
- Jeremy Engleman's Public Textures Hi-Res (doesn't seem to display properly)
- Image*After Free Images Hi-Res
- Free Textures Mid-Res
- Sky Maps
- CG Textures
- Texture Labs

Photos

- <http://Blogpiks.com/> — Free images for blogs, websites and social media
- <http://www.med.harvard.edu/AANLIB/cases/caseNA/pb9.htm> - Brain scan
- <http://www.morguefile.com> - the morgue file
- <http://www.sxc.hu> - Stock exchange
- <http://www.everystockphoto.com> everystockphoto
- http://www.pixelperfectdigital.com/free_stock_photos - Pixel Perfect Digital
- <http://openphoto.net> - OpenPhoto.Net
- <http://www.imageafter.com> - Image*After
- <http://www.aboutpixel.de> - aboutpixel.de
- <http://www.freephotosweb.com/freephotosweb> - Free Photos Web
- <http://www.bigfoto.com> - Big Foto
- <http://www.freephotosbank.com> - FreePhotosBank.com
- <http://www.freejpg.com.ar> - FreeJPEG
- <http://www.stockvault.net> - Stockvault.net
- <http://www.freepixels.com> - FreePixels
- <http://www.freerangestock.com> - Freerange Stock
- <http://pixabay.com/> — requires registration

5.12 Asking for Help

So you've come to learn Blender, eh? You've made a great choice. Blender is one of the most powerful 3D animation and 3D creation tools out there, especially if you're short on cash. Learning how to use Blender can be a daunting task, so **don't give up!** With the help of this wikibook, you can become a Blender power user and put those Maya folks to shame.

If you ever get stuck in a tutorial for some reason, there are a number of places you can turn for help. The best way to get help is with an Internet Relay Chat (IRC) client such as **X-Chat**. Connect to **irc.freenode.net** and talk to blender users in the following channels.

- #blenderwiki
- #blender

- #blenderchat
- #blenderqa
- #gameblender

If you can't get help there, click the "discussion" tab at the top of the page that you're having trouble with, and explain your problem on that page. Wait at least 24 hours for help.

If you're still not getting help, try asking for help in the [BlenderArtists.org forums](#).

5.13 Tips for a Successful Project

The following tips may improve your chances of enjoying a successful project with Blender.

Organize early, organize thoroughly

In Blender projects, you may work with complex scenes containing hundreds of items: models, lights, cameras, scripts, materials, and more. This can be overwhelming if it is not properly organized.

We suggest that you organize as you go, giving each item a descriptive name. There will be ways to search for items by name, so start early. Similarly, give each file a descriptive name and organize files into folders (or directories): a separate folder for each project, with sub-folders for renders, texture images, background images, and so on.

Don't bite off more than you can chew

One of the greatest challenges faced by Blender users is their own imagination. While it's great to be creative and imaginative, you have to pace yourself. Be aware of your limits and the limits of your computer.

When making something from scratch, start with a very simple version of it and add details gradually.

Test frequently; see how well your recent changes work before proceeding to the next step. In this way, you will meet with fewer problems, and the ones you do run into will be simpler and easier to fix because there will be fewer possible causes to consider.

Monitor the complexity of your projects, watching for "lag" or excessive disk activity which might mean you are nearing your computer's limits.

When working on the computer, take frequent breaks to rest your eyes and stretch your muscles. Eat, sleep and exercise regularly. While focused time at the computer is necessary, inspiration and solutions to problems often

come when our bodies are relaxed and our mental focus is elsewhere. While you may be tempted to go wild, keeping a level head while swimming in your imagination will give you a much more satisfying result with fewer headaches along the way.

5.14 Modeling Realistically

Everyone who starts a new career or hobby in 3D Modeling also seems to look around their immediate environment for things to model. Things like your computer monitor, your keyboard, and even the desktop which they are resting are all common “first models.” Developing an eye for geometry takes time and this is a great chance to practice.

For this tips and techniques discussion, I'll start with the concepts of furniture and walls, but the ideas really extend to anything and everything you can model in Blender, from spaceships to fantasy demon-women, from your barbecue deck blueprints to biologically accurate squids.

5.14.1 Assemble Things, Don't Chisel Them

Let's start with the furniture around you.

You may think it's easy to model your desk using a couple of boolean operations on cubes. But it rarely comes out looking quite right. It seems like such a good idea at first. Then you find that the drawers won't open realistically. The corners are all amazingly sharp and crisp. The writing surface seems to be at the wrong scale for thickness. The woodgrain goes in the same direction everywhere. You then realize it will be hard to bevel just the few edges that need beveling. The more you look at it, even with good render settings, the more it looks like dollhouse furniture instead of a real life, full size desk.

Real furniture is not chiseled out of a half-ton cube of mahogany, because trees don't grow that thick, and it's a real waste to chip out the spaces for drawers. Instead, cabinetmakers tend to assemble a piece of furniture from many component parts. Think of those “self-assembly” kits from the local superstore, or even the showroom examples from a nearby unfinished knotty pine store.

Model several kinds of boards and fittings, then copy and assemble them appropriately. You don't need to model every nail and dowel, and you don't even need to know what a dovetail joint is. But you can make realistic furniture when you build things from their natural components.

- Cut lumber in a variety of sizes and thicknesses.
- Measure real example furniture and take note of the construction details.

- Choose varying woodgrains by duplicating materials and adjusting them to be more individual.
- Align those woodgrains or other materials in an appropriate direction and scale for the model.
- Bevel just the parts or edges which are naturally worn or intentionally beveled.
- Any cloth, glass or metal fittings are separate components using their own materials.
- Create an “Empty” type object, give it a descriptive name, and parent all of the pieces to it.

There are a couple more benefits to this form of component modeling:

- Working with things made from multiple real materials is easier: one object, one material.
- The model is more flexible, because they can be re-configured to other styles more easily.
- Components come in handy for other models: adapt a desk's drawers in kitchen cabinets and bedroom furniture.

5.14.2 Separate Objects for each Material



skateboards, modeled in parts

A common question for new modelers is how to make one mesh that uses more than one material. For example, a skateboard with metal trucks, rubber tires, and a wooden deck. While Blender does provide an interface to allow such a combination, it is not usually the best approach for modeling realistically.

If you model the skateboard as separate parts, based on the way different real parts are assembled, then you have a lot more control over the final model. You can replace the wheels or use trucks of different styles later. You can adjust the mesh that defines the shape of the deck, without having to re-assign the areas that should be one material

or the other. And lastly, you can choose to use subsurface tricks on the various parts, while avoiding them on other parts. Even if the whole model needs subsurfaces, it becomes a bit hard to control where one material starts, and another material ends.

5.14.3 Removing the Fourth Wall

Just as with furniture, you may model your first building interior as a hollowed-out cube. It seems straightforward to make a big cube, turn it inside out, and call it a living room. There are a lot of drawbacks to working with a room that was carved out of a single chunk of solid rock.

Buildings and rooms are assembled from parts. Just like the furniture modeling tip, design your rooms as a collection of separate pieces.

- You don't need to measure out the internal beams and joists, but remember they're there.
- Each interior or exterior wall should be its own object or assembly.
- Take care to note the correct dimensions and placement of doorways, windows, ceiling.

The benefits of a component-modeled interior include:

- Easier to see inside, just hide some walls on a different layer.
- Walls are much easier to move around to reconfigure the room layout.
- The same modeling work on one area can be copied to other rooms more easily.
- A more realistic architecture with consistent scaling is achieved quickly.

5.14.4 Proportions and Measurements

For realism, it's vitally important to get the right proportions for familiar objects. If your audience sees a bookshelf with 5" thick shelves, or a human hand which has a 5" long thumb, their brain is going to tell them that something is wrong. They might not be able to decide why, but they'll tell you it looks fake.

Even when you're animating fantasy characters, you should develop an understanding for proportion. Many cartoon characters have very large heads when compared to their bodies; this allows room for greater expression and makes the characters seem younger. Deciding how many "heads high" each character should appear will ensure that your figures all maintain proper proportions to each other and to their environment.

To get good proportions, you need to think about measurements.



soda cans and glass, modeled in parts

- Grab a ruler or find some existing blueprints.
- Think in terms of those measurements for the duration of the project.
- Choose a base scale that will let you work comfortably within a 500x500x500 unit volume.
- Blender limits cameras to 1000 unit distances, for better numerical accuracy.

You can mix and match base units, by using Empty objects as parents. For example, you can model your furniture in inches or centimeters, and model your house in feet or meters. Then when you append your bed.blend's objects, just parent that bed to a new Empty object, and scale the Empty. This will conveniently scale all of the separate component pieces of the object together.

There's only one big wrinkle with working with multiple scales: materials should be designed with a certain base scale in mind. Many material surface properties are scaled according to the units you choose for a given object, or the world units. Just consider this as you design, and check your work critically.

5.15 Modeling tips

This is not really a tutorial, but a list of handy tricks you will probably use a lot when modeling.

1: Extrusion: to extrude, press EKEY in edit mode with the vertices selected, move and click to place the extrusion. You'll basically need this for every model in blender, without it, most models would be really hard to create.

2: Set Smooth: setting smooth will smoothen everything, without increasing the amount of polygons and vertices. You can use it for models with a smooth surface. To apply set smooth, select the mesh, go to the editing buttons (F9) and in the Links and Materials tab, press Set Smooth. If you press the Set Solid button, it will go back to normal.

3: Remove Double: when you remove double vertices, they will melt together, this is good for attaching things to each other. to remove double vertices, select the vertices, go to editing (F9) in edit mode, and in the Mesh tools tab, press Rem Doubt, the limit button next to it is the distance that the vertices have to have to be melted together.

Example use of these techniques:

Extrusion: <http://www.geocities.com/gauravnawani/tutorials/beginner/a-vase-modeling.html> (they explain subsurf here too, but I'd recommend set smooth, because it doesn't increase the number of polygons and subsurf does) http://biorust.com/index.php?page=tutorial_detail&tutid=77&lang=en

Remove Double: http://biorust.com/index.php?page=tutorial_detail&tutid=77&lang=en (same tutorial as in the extrusion section)

Set smooth: <http://www.blender.org/documentation/htmlII/x2681.html>

If you want to add more handy techniques, please add them.

5.16 Cheat the 3D

5.16.1 Ways to Improve Performance

In 3D Graphic Design, there are many issues to consider. First, consider how the project moves, how the project performs. Next, consider how you create objects in your project, especially when working with meshes and lighting.

Mesh

Modifying an object's properties

When you add a modifying property to an object, the computer has to calculate that property every time it moves in the animation. These modifying properties include soft bodies, particles, or mirror textures. They can significantly slow down the computer.

To make things move faster, pre-calculate and “bake” the modifier into the object. To bake a modifier, such as a soft body movement, save the modifier as a permanent animation. That way no matter how the object changes, it moves the same way. As a result, your computer does not calculate how the object moves. The move is already set.

There is a drawback. If the object moves and hits another section with different slopes, it will continue to move as if it was in the original baking site. You will have to calculate and bake the section again to get new results.

Making 2D backgrounds

Scenes will have a foreground, background, sky, and more.

Your computer will take more time to render a scene that is big, open, and filled with modeled objects. This wastes time and memory, especially when modeling an object far in the distance. I'm guessing however, you do not want to make a quick, plain model. So what do you do?

That is easy. You make a 2D background.

- Put the distant image or background on a plane.
- Verify that everything on the plane around the image is black.

If it is not black, set the Alpha to 0 if the image was saved in a format that supports Alpha. This will retain the detail but reduce the number of faces the computer must count in the scene.

- Set the image to Billboard to prevent the image from skewing by perspective.

Materials

Using UV maps

For those who do not know the term, a UV map is an image applied to a 2D plane. The 2D plane is then applied to the mesh.

Here is how to create a UV map:

- Spread out the image onto a 2D plane.
- Apply the 2D plane over the face of the mesh using the 3D view.

5.17 Performance vs. Quality

Mesh

Polygon Count - when you work with meshes in 3D, be aware that the memory usage in Blender with reference to meshes comes from the number of faces (or vertices) the mesh contains. That means that if you have many useless faces (such as 55 faces on the side of a mesh that are as flat as one face), the number of faces will directly affect the Render time of Images and Animations. Also remember that for smooth rounded meshes, you do not need an infinite number of faces to make it look really nice and smooth. You actually don't need that many faces to keep a smooth slope looking smooth when viewed from the side. The viewer's eye will not actually identify every flat face, but will trace it as a smooth

curve.

Face Structure - 4-side (quad) vs.3-side (tri) - Poly Count is very important when working in 3D graphics, and depending on the type of mesh you are making you could be wasting a vast number of faces because they are not the right structure. For example, if you want a smooth object, you can make it look mostly smooth when viewed from the side if you make it with well-structured 3-vertex faces. You will also yield the same results with fewer faces than if you use 4-vertex faces. Another example is that if you need to apply UVTexture, or have something to animate, a mesh being evenly deformed will yield a lower face-count if it is a 4-vertex versus a 3-vertex face-structure.

It is worth bearing in mind, however, that most graphics libraries deal exclusively with triangles. This means if you are planning to export your mesh to a game or other external 3D application, whether you use quads or triangles will make no difference as all quads will likely be converted into two triangles anyway.

Lighting

Using RayShadows or Buffer Shadows - There are a few options you have for using shadows. Ray Shadows use an advanced algorithm to trace the edge of any interfering objects and create a perfect shadow onto the receiving object(s); however, the Ray Shadow calculation is memory-intensive and can seriously slow down your Render-Time. Buffer Shadows, on the other hand, use a different algorithm for similar results. The difference is that buffer shadows use a bit-rate of shadow “pixels” that fill in the shadowing area. You can adjust the bit-rate to make the shadow higher or lower quality. This calculation is much more memory-friendly, and your Render-Time will not jump up as much as with Ray Shadows. An interesting topic because this will change with better hardware performance and eventually won’t matter. Until we get into 3D.

5.18 Modeling a Gingerbread Man

In this tutorial you will learn how to make a simple gingerbread man. In a later tutorial you will be able to make an animation with this gingerbread man.

In this tutorial we will tie together everything we've talked about up to this point, including extruding, subdividing and rendering, and throw in basic lighting.

5.18.1 Modeling

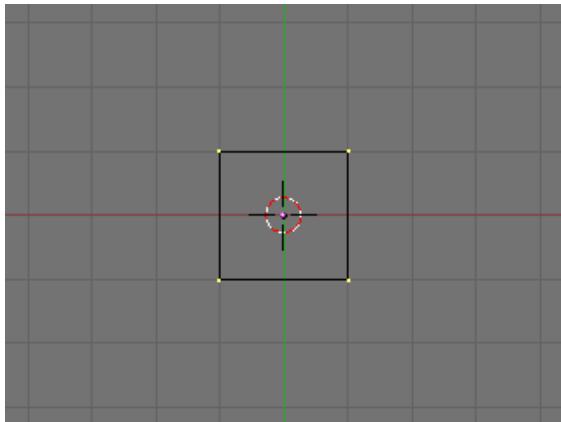
First, start Blender. You should see the usual top view of a cube in the 3D window, surrounded by a grid ‘floor’.

Review— Zoom in or out with **SCROLL** or **CTRL+MMB**. Pan with **SHIFT+SCROLL** and **CTRL+SCROLL**. Make sure you are in orthographic view. Press **NUM5** to toggle between orthographic view and perspective view. You can tell you’re looking at an orthographic view while looking down from the top (**NUM7**) of the cube: wherever you pan the window, you never see any part of the cube except the top.

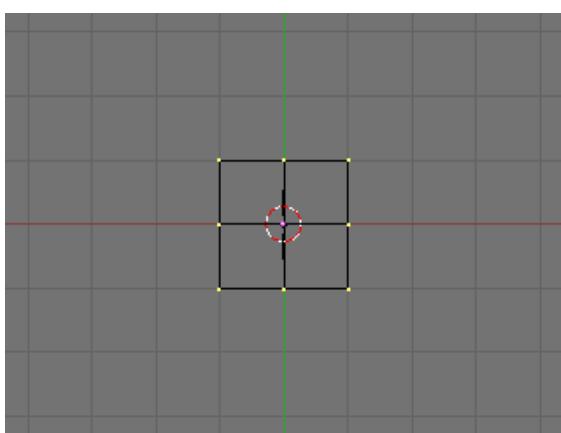
- Display editing controls in the buttons window by clicking on the Editing context button  or by pressing **F9**.

Adding Vertices

- Select the cube by clicking **RMB** on it. (It may already be selected.) Remember, selected objects are outlined in pink.
- Switch from object mode to edit mode by pressing **TAB**, which toggles between object and edit modes. In edit mode, at first, you’ll see colored dots at the vertices of the cube. Selected vertices are highlighted in yellow. Unselected vertices are pink. (You’ll see the dots when editing in vertex select mode. In edge select or face select mode, you’ll see colored edges and faces instead.)
- Select all vertices of the cube. Press **AKEY** once. If the vertices are yellow, you’ve selected all of them. If they are pink, you’ve deselected all of them. If necessary, press **AKEY** again so that all vertices are selected/yellow.
- Subdivide the faces of the cube, using any of the following methods. (All vertices of the cube should still be selected.)
 - Click the Subdivide button in the Mesh Tools panel in the buttons window.
 - With the mouse pointer in the 3D window –
 - Press **WKEY** to display the Specials menu, then choose Subdivide.
 - Press **SPACE** to display the toolbox, then choose Edit → Edges → Subdivide.



- Click the 3D window's Mesh menu, and choose Edges → Subdivide.
- Your cube now has more vertices. **Subdividing** edges adds vertices so you can create more complex shapes.

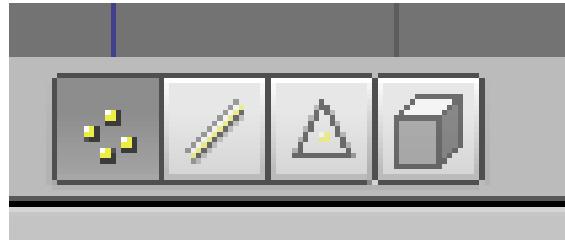


Selecting a Subset of Vertices

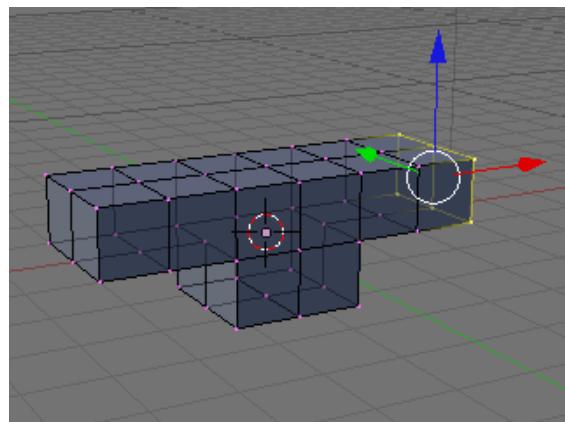
We'll select six vertices on the upper half of one side of our cube, where we'll extrude an arm.

- Deselect all vertices by pressing **AKEY**.
- View the cube from the front (**NUM1**).
- Select the six vertices on the top half of the left side of the cube (see below) using one of the following methods.
 - Press **BKEY** and drag a rectangle around the top left and middle left vertices (viewing from the front).
 - Press **CKEY** to see a circle around your mouse pointer. **SCROLL** to change the size of the circle. Position the circle around the top left and middle left vertices, together or one at a time, and click **LMB**. Click **RMB** to finish.

- Take a closer look at the selected vertices by viewing the model from a different angle (drag with **MMB**).
- If you find that you have only selected two vertices and not six, make sure the “Occlude Background Geometry” button is off. That button is the right-most of the selection mode buttons, below. (It's called “Limit selection to visible” in Blender 2.45 and earlier. It doesn't appear if Blender is drawing in wireframe style.) Try selecting the vertices again.



- If you still selected just two vertices, change to a wireframe drawing by pressing **ZKEY**, which toggles between wireframe and solid drawing types. Try selecting again.
- Yet another way to select the six vertices is to select the two faces they define. Click the Face Select button from the selection mode buttons (above), or press **CTRL+TAB** to display a Select Mode menu and choose Faces. Rotate the cube to view the left side and click **RMB** in the center of one of the two upper faces. Hold **SHIFT** and click **RMB** in the center of the other upper face.

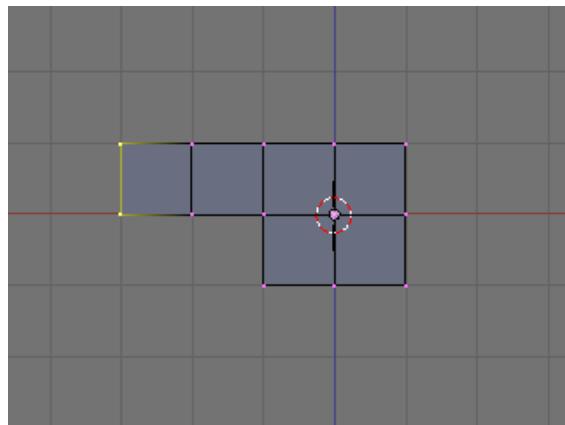


Extruding Arms

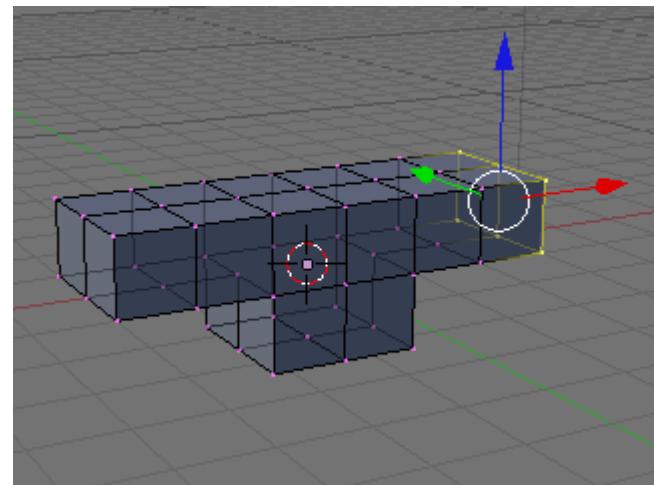
- View from the front by pressing **NUM1**.
- Extrude the selected vertices/faces by pressing **EKEY** to display an Extrude menu, then choosing Region. Hold **CTRL** to snap your position to the

grid, then move your mouse left to put the new vertices on the adjacent gray line of the grid one unit to the left. Click **LMB**.

- Repeat so that your model looks like below (from front view, **NUM1**).

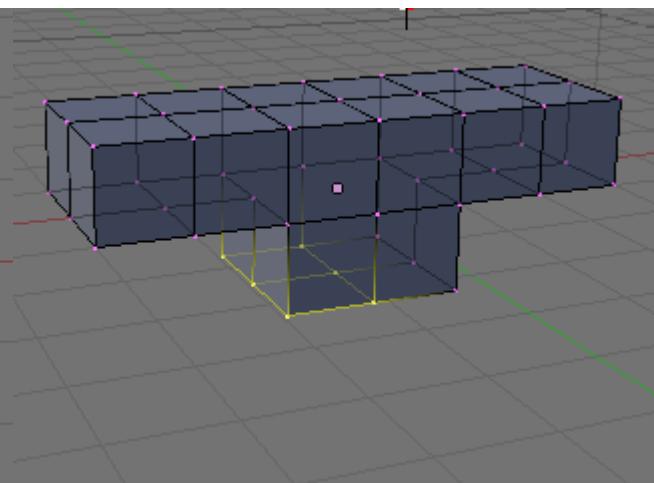
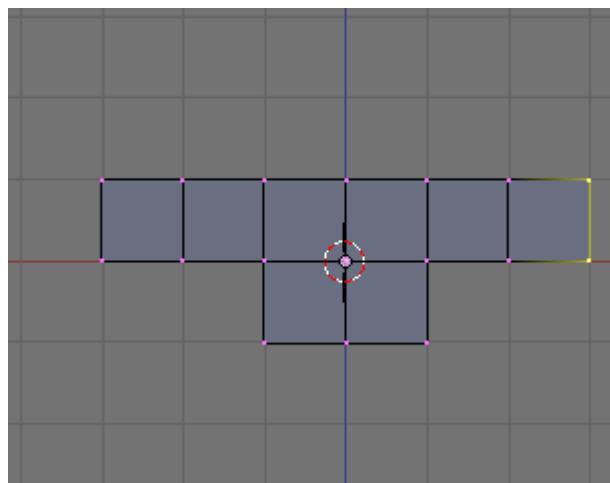


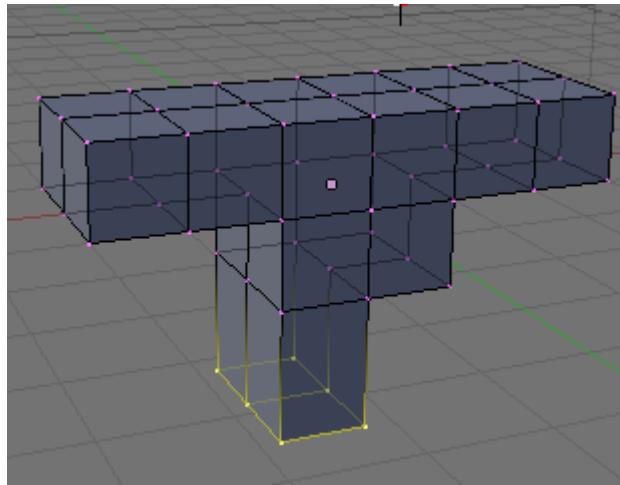
- Deselect all vertices/faces with **AKEY**.
- Perform the same extrusion on the right side of the cube, selecting six vertices and extruding twice as explained above.
- The gingerbread man's arms are in place, as in the illustrations below.



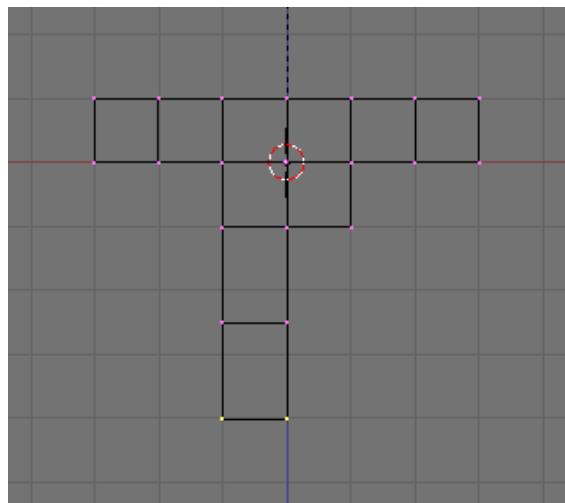
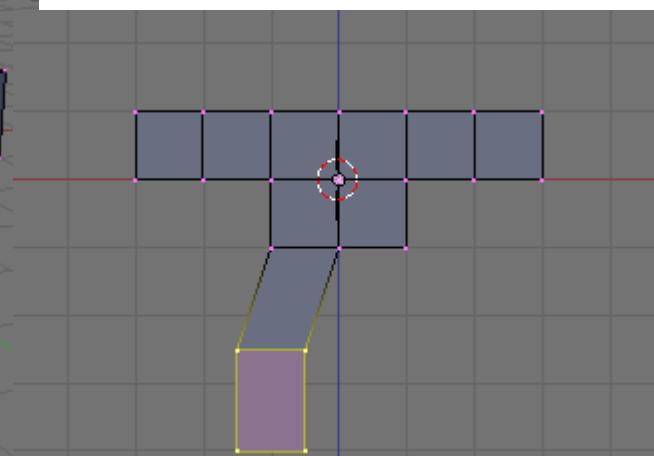
Extruding Legs

- Make sure all vertices are unselected, using **AKEY** once or twice.
- View from the front (**NUM1**).
- Select the six vertices on the left half of the cube's bottom. Extrude them downward to a point in between the first and second heavier gray lines beneath the cube. (The gray lines in the grid represent Blender units.) Holding down **CTRL**, the extruded region may snap to the heavier gray lines—hold down **CTRL+SHIFT** and the region will snap to tenths of Blender units. Click **LMB** to finish.
- An alternative to positioning the extrusion with the mouse is simply typing the distance. Enter 1.5 to extrude 1 1/2 units out. On a Mac, enter the number 1, press fn with the key that is right under **LKEY** and **MKEY** on Azerty (the one with /:), and press the number 5.

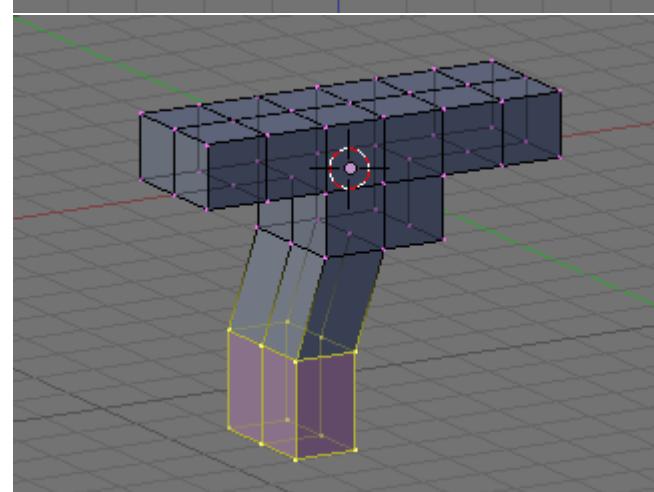




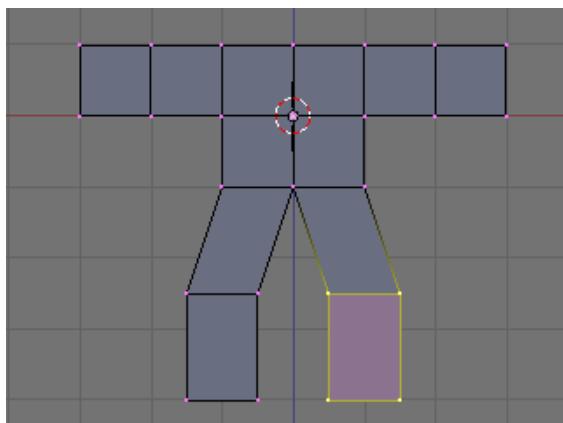
- Extrude the same region again, to the third gray line (1.5 again). It should look like this:



- Shift the lower leg to the outside as follows.



- Create a second leg on the right side, in the same fashion.



- Select the bottom 12 vertices of the leg (which look like 4 from front view), using **BKEY**.

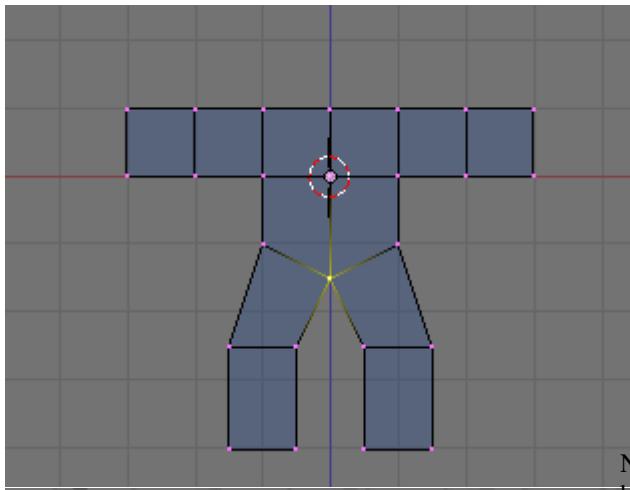
- Grab the selection with **GKEY**. Press **XKEY** to limit movement to the x axis. Move the vertices to the left by half a square, holding **CTRL** or **CTRL+SHIFT** to snap to the grid, and click **LMB**.

Dropping the Groin

Lengthen the groin (where the two legs join):

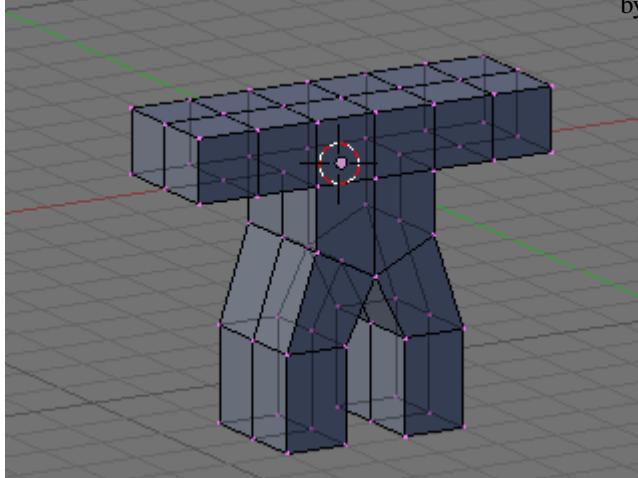
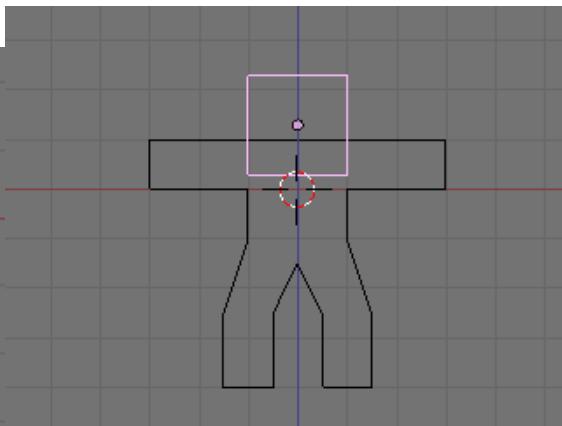
- Deselect all vertices by pressing **AKEY**.

- Select the 3 vertices at the groin (which look like 1 from front view), using **BKEY**.
- Grab the vertices by pressing **GKEY**. Press **ZKEY** to limit movement to the z axis. Move the vertices down 1/2 of a square, or type **-0.5** to specify the distance. (Older versions of Blender require **NKEY** before typing **-0.5**.)



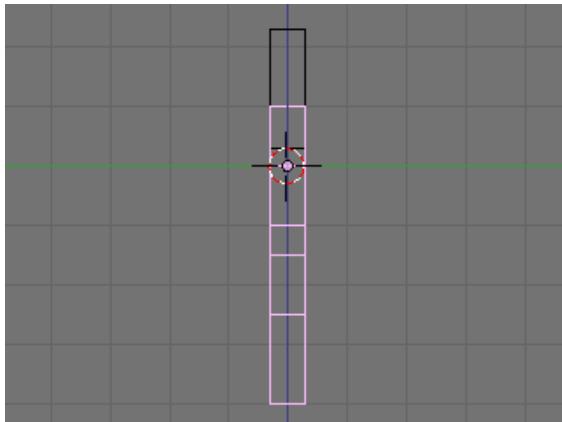
hit **SPACE** and , in the menu that comes up, choose **Add → Mesh → Cube**.

- Press **GKEY** and put your new cube about 1/3 of the way down the neck (to achieve this, you can press **GKEY** and **ZKEY** : enter **1.33**).



Now we will make it look more like a ginger bread man by making it thinner.

- Select all with **AKEY**.
- Go to side view with **NUM3**.
- Press **SKEY** for scale and press **YKEY** for Y-axis and then move your mouse to the middle until it is about **0.3** (use **CTRL** for fixed values).
- Remember X-axis is the Red arrow/line, Y-axis is the Green one, and Z-axis is Blue (like RGB video mode).



Adding a Head

- Switch the 3D window from Edit mode back to Object mode by pressing **TAB**.

Noob Note: I found it easier to remain in edit mode while creating the cube as this makes the cube part of the original mesh. This allows you to add the subsurf to all parts at once.

- Click **RMB** on the object to select it then press **SHIFT + SKEY** and select **Cursor → Selection**. This will make sure the cube you'll add next will be near where you want it.
- Press **SPACE** and put your mouse on the mesh option and select cube. In others versions, you can also

- Use the **MMB** to spin the view around and examine your handiwork.

At this point, it doesn't look entirely like a gingerbread man, does it? It's a bit too ... chunky. For the last bit, we'll smooth it out.

- Make sure you've selected the body in object mode.
- Select the editing panel in the buttons window (or hit **F9**).
- In the Modifiers tab, Add a "Subsurf" modifier.
- Set the level of the subdivisions to 2, and the number of render levels to 3.

Noob question: When I add the cube for the head, it stops me from being able to edit the body - it will only select the head to apply subsurf to, even if the body looks like it's selected!

Answer: When you created the cube you made a second object. To select a different object, press tab to enter Object mode. Select the body. Then enter edit mode again if you want to edit the body.

- You can press the **ZKEY** to switch back and forth between wire-frame view and solid view.
- (Noob Note: Easiest way to really get a feel for what is going on in the 3d world is to split into four screens and setting each one to **NUM7**, **NUM3**, **NUM1**, and **NUM0** to see all angles and what it will look like at render.)

Noob Question: How?

Answer: To split an area move the cursor to an area between two current areas (e.g. between the 3D view and the buttons), when you see the double ended arrow (used to move the divide) click RMB and select *Split Area*, you will then see a line appear dividing the area in two. Move this to where you want the divide and click LMB.

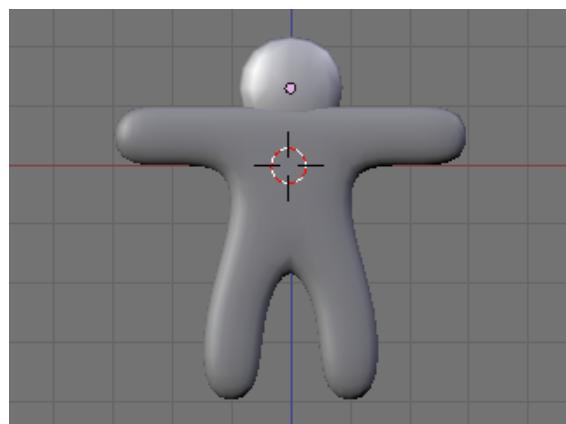
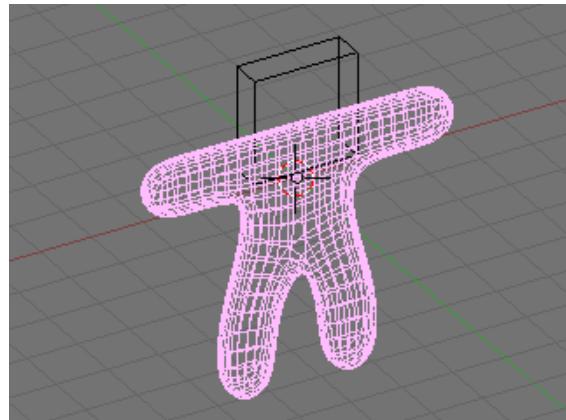
- In the 'Link and Materials' section, select 'Set Smooth'.

Noob question: Where? Assuming this refers to the 'Materials' section on the 'Properties' window, there is no 'Smooth' setting.

(Note that here I had the same problem as before, with superposed vertices. Select all vertices, then press **WKEY** and select Remove Doubles to clean your model. You will see that it will look much better after removing the extra vertices with Remove Doubles)

- Press the **ZKEY** to return to wire-frame view.
- Now repeat the process above to smooth the head.

Looks a lot more like a gingerbread man now, doesn't it?



5.18.2 Camera Positioning and Rendering

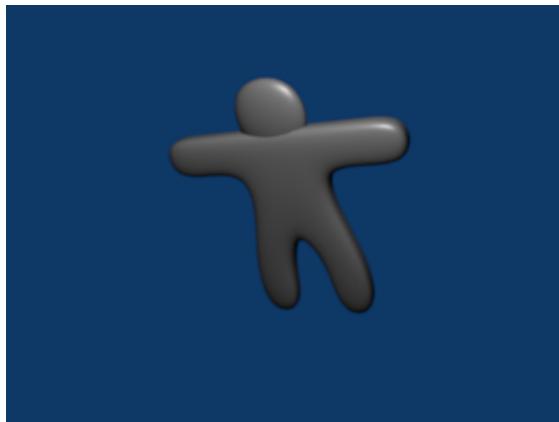
This guide will show you how to intuitively get the best frame of your 3D scene with no effort!

- Press **TAB** for Object view mode.
- Press **NUM0** to get the Camera View.
- Select the camera by clicking **RMB** on the outermost rectangle.
- Press **GKEY** and move your mouse to adjust the position of the camera (**XKEY**, **YKEY**, **ZKEY** and **CTRL** may be useful here).
- In addition, you can press **NUM7** to get the Top View and press **RKEY** to rotate the camera to the best angle.
- After you are happy with the position, press **F12** to render it.

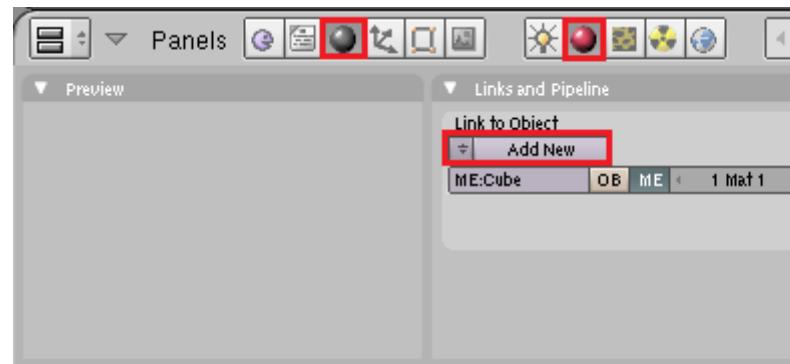
If your render comes out a little dark, try moving the lamp closer to the gingerbread man.

Noob note: Another way to move around the camera is pressing **SHIFT + FKEY** after pressing **NUM0** to enter Fly mode. The keys for fly mode appear in the header of the 3D view pane.

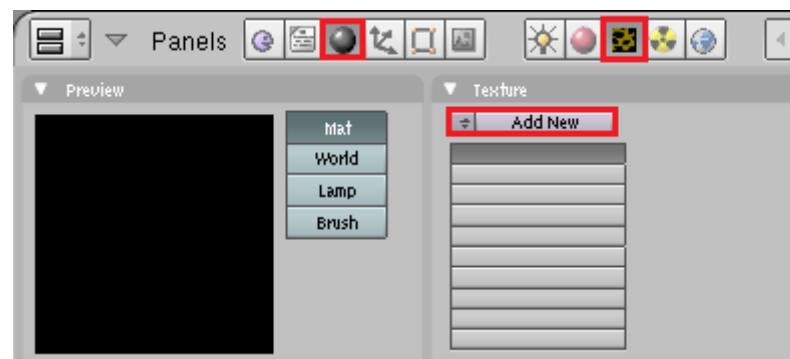
Noob note: **Ctrl+Alt+NUM0** “teleports” the camera to your 3d view.



Noob Note: By pressing X, Y or Z twice you will use a local base of the space, with those it's much easier. For example if you are facing the Z axis from 45 degrees, and you want to go left 1 unit, using the global base, you will have to go 1.72 (around $\sqrt{2}$) along X and the same along Y, instead moving by 1 in the local frame of reference.



- Press **F6** to open the “Texture Buttons” panel or use the textures button.
- In the “Texture” panel, click “Add New.”



- Change the “Texture Type” to “Stucci.”



- In “Object Mode,” select the body (or the head.)
- Press **F5** to open the shading panel or use the shading panel button.
- In the “Links and Pipeline” panel, under “Link to Object,” click “Add New.”

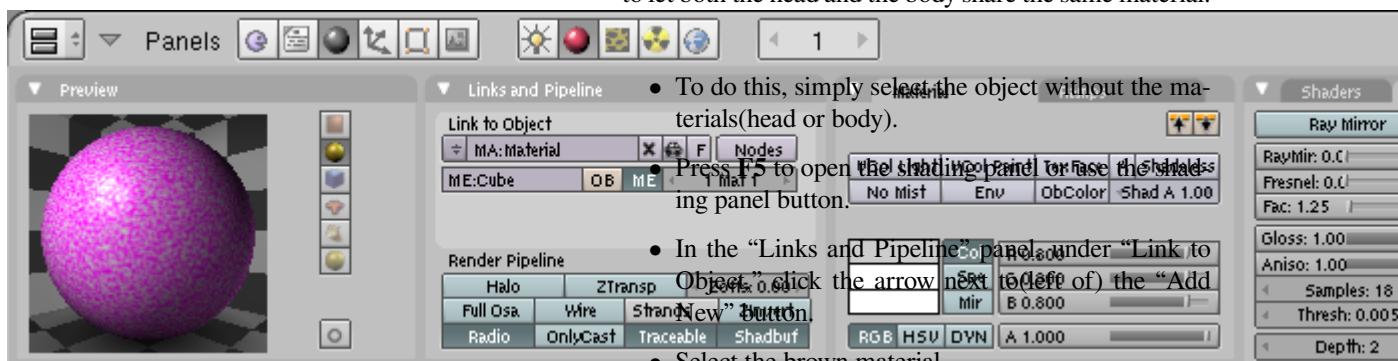
- In the new “Stucci” panel, change “Noise Size” to something near 0.025 and leave the “Turbulence” at 5.00.



Note: When finished with this section of the guide, come back to this panel and try different combinations of “Plastic,” “Wall In,” “Wall Out,” and “Soft Noise” / “Hard Noise.” Press F12 to render after each change to see the effect.

- Press F5 again or use the “Material” button directly on the left of the “Texture” button. Then look for the “Map To” panel.

Note: While it is true that textures can only be applied to one object at a time, textures as well as materials can be shared between objects. In this case it is best to let both the head and the body share the same material.

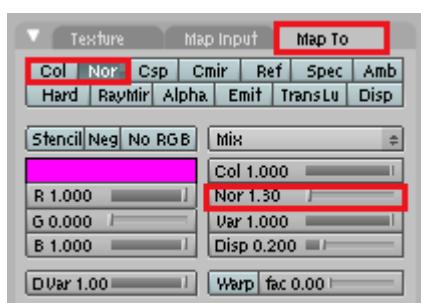


- In the “Map To” panel, deselect “Col/or” and select “Nor/mal.” and change the “Nor” Value to approximately 1.30.
- In the “Map Input” panel, change the texture coordinates to “Object” by clicking the corresponding button.
- In the “Material” panel, change the “R/ed” slider to approximately 0.400 and the “G/green” slider to approximately half that, about 0.200. Blue can be set at 0.00.

The steps in this section give a nicely textured, brown surface to the “Gingerbread.”



Now all you need to do is add eyes and gumdrop buttons!



5.19 Modeling a simple space-ship

5.19.1 Modeling a simple spaceship using Blender 2.49b

This is a simple tutorial on how to build a simple spaceship using blender 2.49b. One of the best things about blender is the adaptability that can be accessed through its user interface. This tutorial will give you a look at some of the included blender scripts. If you would like to make your own, however, you can refer to the scripting chapter of this wiki book. So now, I guess we can get started.

Step 1: First, open blender. You should get the default blender scene with the cube in the center. If you don't, try resetting your blender settings. Select the box in the center of the screen.

Step 2: Now we're going to shape the box into any shape we want. For now, we are going to shape it into a sort of triangle. If you want to use it as a box, that's fine, but I suggest following this tutorial using it as a sort-of triangle. To be able to modify it, in the lower buttons area (with stuff like Modifiers, Shapes, Multires, Mesh, and Link and Materials) you should see a drop-down menu that is currently on Object Mode. Change that to Edit Mode or hit the **tab** button. Now you will be able to modify different aspects of your box.

Step 3: With your box selected, hit **Ctrl+Tab**, and choose Faces. This allows you to modify the different faces. Faces are basically the different sides of your object, Edges are the edges of your object, and Vertices are points on your object where edges connect.

Step 4: Now, on your box, right click on one of the sides to select that side. If you want, first take a look around the box and then choose a side, but don't select the bottom or top face. To rotate around the box, hit the 8, 2, 4 or 6 button on your number pad, the one to the right of the arrow keys usually. Those buttons allow you to rotate around your objects and change your view. After you select the face you want, then rotate so that the face is facing towards you and looks like a square, and so that the grid around it looks like a flat horizontal line.

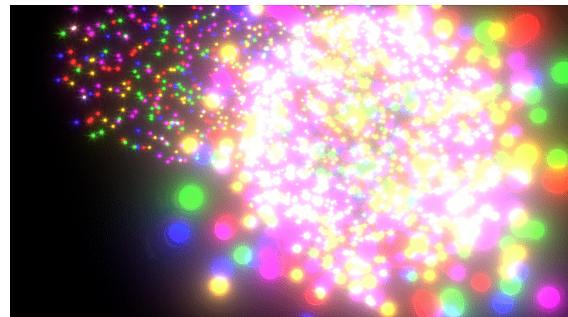
Step 5: There should be three arrows converging in the center of your face (or in the center of your box), and either the red or the green one should now be in the center of your face, and the other should be going off to the left or right, with the blue arrow going straight up or straight down. Use the 4 or 6 number pad button to rotate your view so you can see whichever arrow was in the center of your face, and left click mouse button on it, and hold the left click button, and drag the arrow around. Drag it until you have a rectangle. <http://i238.photobucket.com/albums/ff55/bryguy336/uh.png> <http://i238.photobucket.com/albums/ff55/bryguy336/box.png>

Step 6: Now to resize that face. With that face still selected, hit the **S** key, and move the mouse around. This should enable you to change the size of the face, and once you click, it should stop resizing and stay at that size. Experiment and get the hang of it, and then resize it so that it is an itsy bitsy teeny tiny box, and your rectangle more like a triangle. Congratulations! You have just made your first shape/object

5.20 GIMP)

This is my first tutorial. I'll be teaching you to create this

The quality is low because it is a gif. if it's a png or avi the quality'll be much better.



Created for a tutorial for Blender particle systems

5.21 Part 1 - Preparing the Scene

5.21.1 Start a new scene in Blender

- Delete the default cube.

5.21.2 Prepare the individual particles

Go to layer 2 -

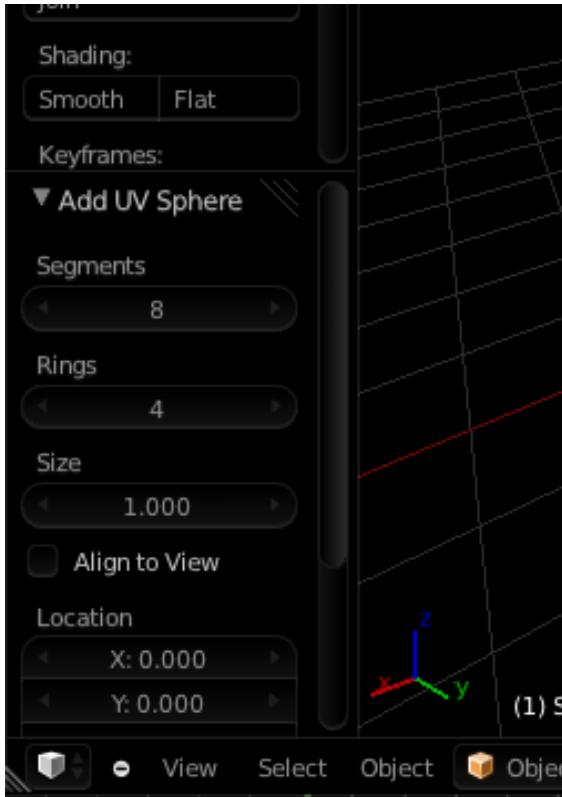


The layers buttons

Add a UV Sphere with 8 segments and 6 rings(Go to the toolshelf and change the values)

Add a new material with any bright color. Change the emit value to 2 or higher -

Note - if you want a single-coloured version, you can skip a few of the following steps -

*The Toolshelf*

Duplication of the particles

Duplicate the sphere and move it nearby (Shift + D). Duplicate the material by clicking on the 'plus' button indicated in the picture below. Change the color to another bright color

Important note - Not clicking the plus button will change the color of the previous sphere also. Repeat this process until you are satisfied with the no. of spheres with different colours. Now create a group (Ctrl + G) Name it anything you like in the toolshelf (T toggles the toolshelf).

Now you're ready to start creating the animation and particles!

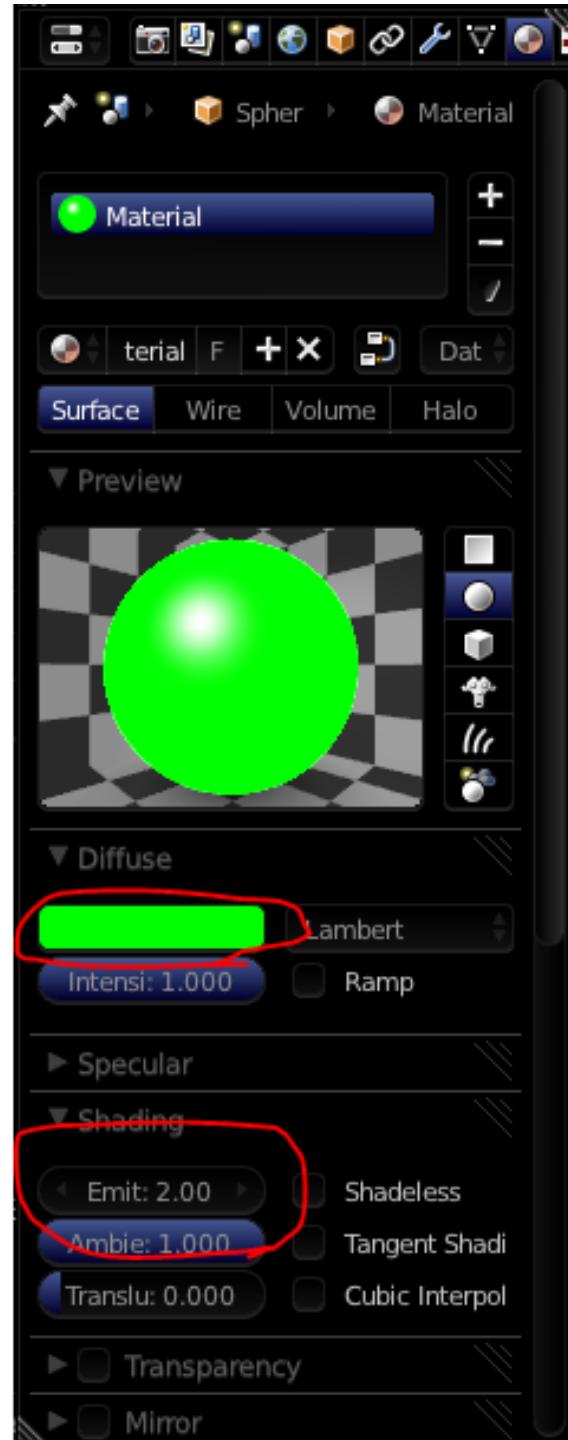
5.21.3 Defining the motion of the particles

Go back to layer 1.

Creating the path for the particles to follow

If you haven't learnt about Bezier curves already, go ahead and do so now.

Add a bezier curve, and tab into edit mode. Now you have to model the path by extruding and subdividing the curve. The shape of the curve defines the path of the particles. After you are done with the curve, you can proceed to

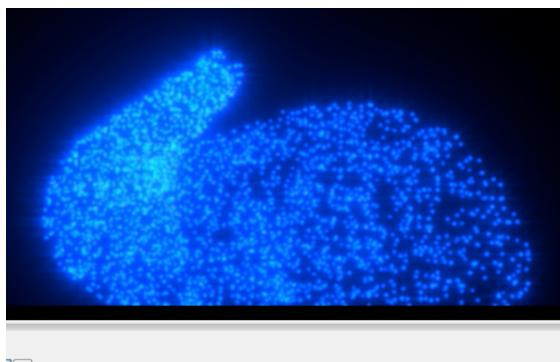
*Material settings*

create the emission object.

Creating the 'emitter' of the particles

Add a UV Sphere with 32 segments and 16 rings. This is going to be the emitter, i.e. the object that gives out those coloured spheres.

This is going to have a completely transparent material

*Material settings**Single-coloured version*

since hiding it hides the whole thing.

So go to material properties, add a new material. Scroll down to the Transparency panel, check it and change the alpha value to zero, making it completely transparent. Also change the specular intensity to zero. Leave everything else as such.

Making the sphere follow the path

In this part, you are going to make the sphere follow the curve you had modeled earlier. Select the cube, and add a Follow Path Constraint. Select the target, i.e. the curve and click on Animate Path. All these steps speak for themselves, there's no need for me to explain. You could check follow curve, but since this is a sphere this is not needed. All it does is make the object face the direction it is moving towards.

As for the speed of the object, you can change it by -

- Select the curve.
- Go to Object properties -
- Scroll down to Path Animation panel.
- Change Path Animation Frames value to the no. of frames you want the animation to last. I prefer it to be 150.

*The Object Properties*

In the end, make a few changes until you are satisfied with the result. Now you are ready to proceed to part 2.

5.22 Creating Weapons based on 2D Images

Alright, this is my first tutorial here, I made this a while back, but hopefully it will be a good addition to this Wikibook. It was originally made for a game called **Fable - The Lost Chapters**, but I think it is worthy of another place in the world. Please note that this is a WIP and will be updated time to time.

5.22.1 Modeling Technique 1

Modeling Technique 1 Part 1 Video

Part 1:

1. After emptying the screen, go to view and click background image, load, then choose a picture of the weapon you want
2. Go to top view and add a square, then delete two of the vertices and place one of the remaining on a point on the picture, and the other out of the way.
3. Select both vertices and subdivide.
4. Move the generated point to the next good-looking spot on the picture
5. With that selected still, select the outside vertex also, and subdivide.

(Note: A much easier way is to select one vertex, move it to where you want it, select the other vertex, move it to the next spot, then extrude the second vertex to the next point, rinse and repeat. This requires a lot less effort.) Another guy's note: For an easier way, just click **CTRL+LMB** where you want the next point to appear.

1. Continue until you are on your third to last one, then select your outside vertex and move it where you would move the next vertex.
2. Select the first and last vertices and go to **mesh>>make edge/face**
3. Extrude to half your preferred thickness, then extrude the rest of the way.

4. Select the center vertices on the blade, and scale up.

Modeling Technique 1 Part 2

Part 2:

cont.)

1. Then move vertices to your liking.
2. Select the vertices of the handle, go to mesh>>vertices>>separate.
3. Add a modifier, choose subSurf, turn up the level until it looks close to what you want without too many vertices. Click apply next to the modifier.
4. Modify vertices (using proportional edit helps) to your liking.
5. Go to object mode, and turn off double-sided on any meshes that are.
6. If black appears on any parts of the mesh, highlight it, go to edit mode, select all vertices, go to mesh>>normals>>recalculate outside.
7. If there is still black, select those faces and go to mesh>>normals>>flip.
8. If there is still black, then you are missing a piece of mesh. Highlight the vertices around the hole, go to mesh>>vertices>>fill.
9. If black did not appear, then select everything (in object mode) and go to object>>join objects and say yes.

5.22.2 Modeling Technique 2

[Video Here](#) Download it for high quality.

Blade:

1. Select all with the 'A' key, and delete everything.
2. Go to view and choose background image >> use background image >>load and pick the sword you'll be doing.
3. Close the dialog after you see it in the background (you can change the brightness of it by changing the 'blend' option)
4. Make a bezier circle by pressing spacebar and going to add>>bezier>>bezier circle.
5. put a bar at each main point of your weapon (anywhere the curve changes direction)

Bezier Controls

1. Modify the bars until they match.

2. You might want to go into wireframe view.
3. Go down and turn up the bevel depth a little, to give it some sharpness.
4. To lower the poly count you should turn down the DefResolIU number.
5. Go back to object mode, hit spacebar, go to object>>convert object type>>to mesh.
6. Modify vertices to your liking.

For the handle:

1. Do steps 1-7.
2. Turn up the bevel resolution, to give it some roundness.
3. Repeat steps 8-10

For That one thing (see video):

Just watch the video, but here is a summary.

Extrude and flatten a circle, use proportional edit to make it easier on you when you curve the circle.

For hilt guard:

1. Repeat steps 1-6
2. Add some extrusion.
3. Repeat steps 8-10.

5.23 Modeling with Meta Balls

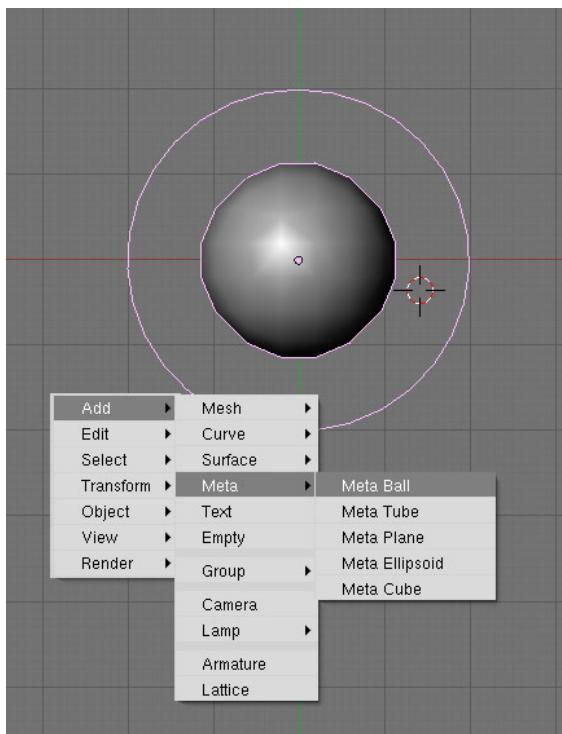
5.23.1 Getting started

To begin, open a new scene in Blender. Let's clean up the scene a little by selecting the lamp and camera in the scene, press M to move them to a different layer and click on the fifth layer from the left to place the objects. Go back to layer 1 and delete anything else. You may also want to get rid of the grid by opening the View Properties and turning off the Grid Floor and X & Y axes.

To block out our character, we're going to use an object type that is probably the least used and useful of any known to mankind. Let's hear it for... Metaballs!

5.23.2 Sculpting With “Lumps of Clay”

Go into the top view (important), press Space and add a **metaball**. Metaballs are a nifty, ancient piece of 3D technology that is useful for creating blobs. (Similar to lumps of clay, eh?) You create simple primitive shapes and scale and rotate them to block out your character's



shape. When the primitives come close to one another, they “bleed into one another” in much the same way that water droplets merge when they touch. Cool.

If you're using an earlier version of Blender that jumps out of Object mode into Edit mode when you create an object, then press Tab to switch back to Object mode, as you won't be able to scale your Meta primitive non-proportionally in Edit mode.

In Object mode, you can change these options for the entire Meta object, while tabbing into Edit mode gives you more options for the selected meta primitive, such as changing the type from Ball to Tube, Plane, Cube, ;etc.; (You can also make “Negative metaballs.”)

Press **Shift+D** to duplicate the Metaball, and place it where you like. Continue blocking out your character, building enough blobs to represent the limbs or forms you will need to sculpt your masterpiece.

The balls at the end of the limbs, were scaled **SKEY** larger, and then moved **GKEY** out a bit more.

Don't get carried away and put in too much detail at this stage: use as few shapes as you need. (This is supposed to be quick and fun, after all...)

5.23.3 Meta-mess!

You should still be in object mode.

Now that you've got something that resembles what you're after, select all the Metaballs and (be in object mode) type **ALT+C** -> “delete original”, to convert it

to polygons so you can actually do something with your blob.

Noob note: if you pick “keep original” you will still have the meta balls present, plus have a mesh version of the metaballs sharing exactly the same space. When you select “delete original”, the meta balls are turned into the mesh, and the circles that the metaballs were originally in hang around, but are empty.

Still in object mode!

Delete any of those black rings left over from the metas and select your new polygon mesh.

If you Tab into Edit mode you will see terrifying ugliness instead of nicely gridded mesh. “Surely we can't be expected to create anything useful out of this!” you shriek. Take it easy, my friend. It's time to add a *Decimate* modifier (make sure you're in object mode when doing this to see changes).

Switch back to object mode!

The **Decimate** modifier (you will only see things change if you're in object mode) will do two things for us. Its primary job is to *reduce* the poly count of a mesh. A pleasant side-effect for our purposes is that it will begin to *rearrange* the topology into a more manageable heap of triangles and quads. Keep reducing the Ratio slider below 0.5 until it becomes as coarse as you can stand. You want the lowest polygon base you can have that still maintains enough detail in the limbs and shapes you made with the Metaball phase.

One thing to watch for is that this process sometimes creates holes as it does its best to simplify the mesh. I find that usually you can slightly change the Ratio to fix the problem, but if you're still finding holes, check out the “Tips & Tricks” section at the end.

As you can see, this step greatly reduces the Face Count, which will be good later. Next we need to get rid of as many of those triangles as possible. Click **Apply** (in the modifier panel)

Switch to Edit mode!

Press A until all of the vertices are selected (turn yellow) and hit Alt-J to convert the faces from triangles to quads, which will subdivide better. Now you should have something you can work with.

5.23.4 Beginning to sculpt

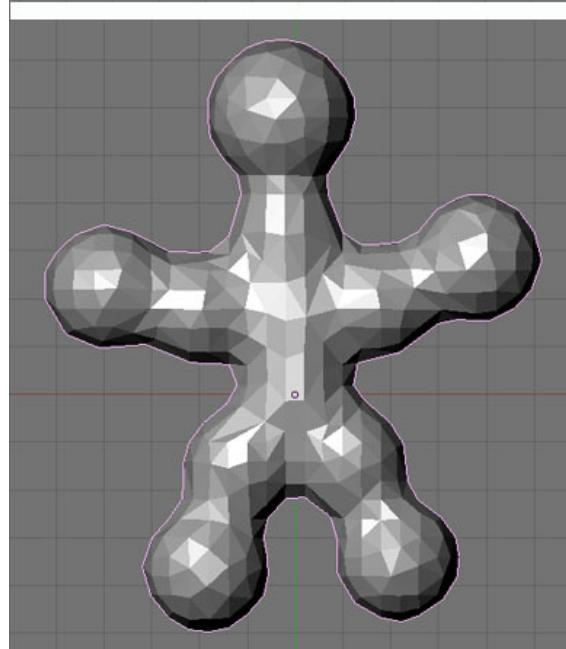
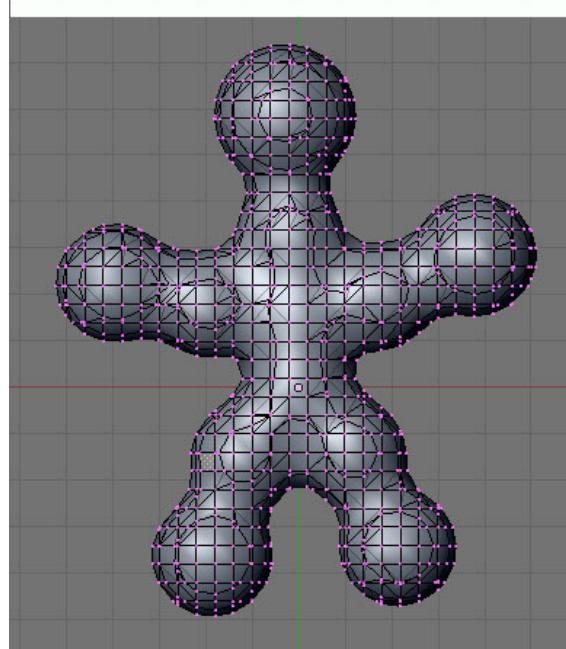
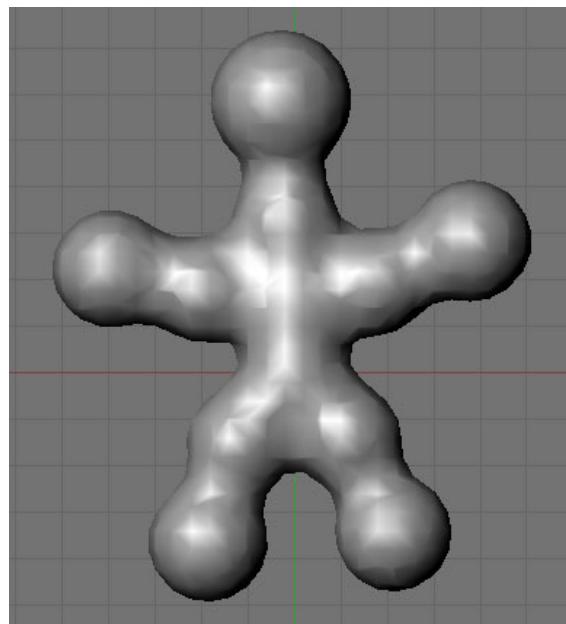
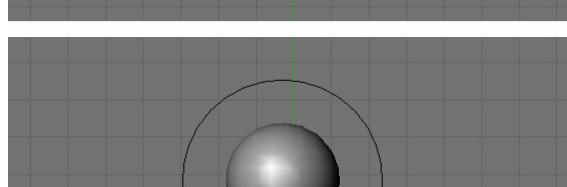
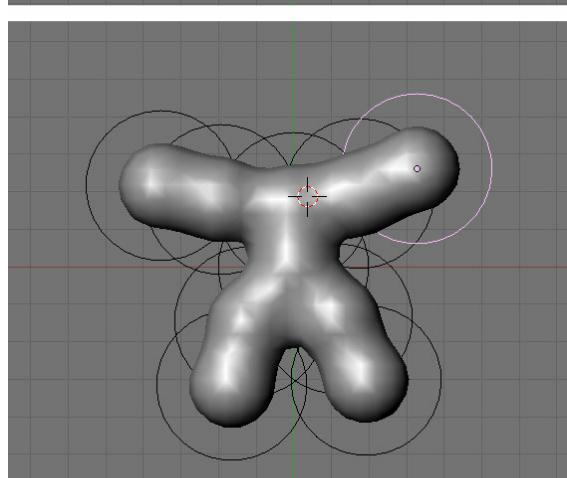
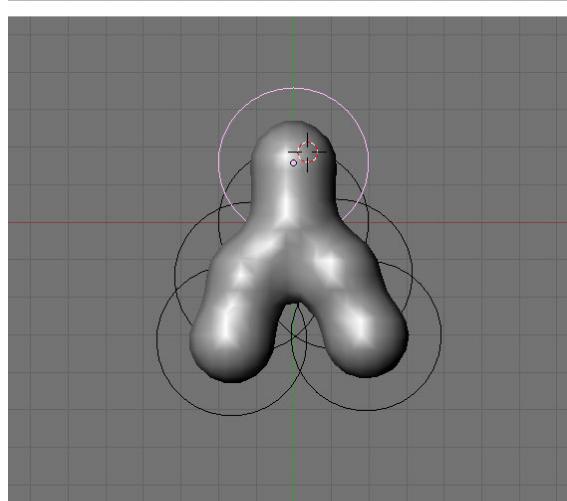
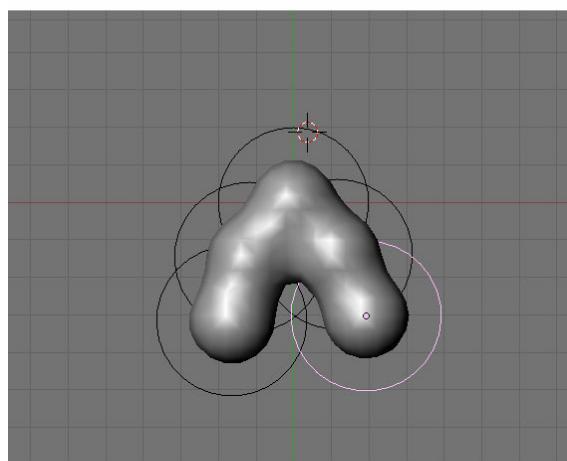
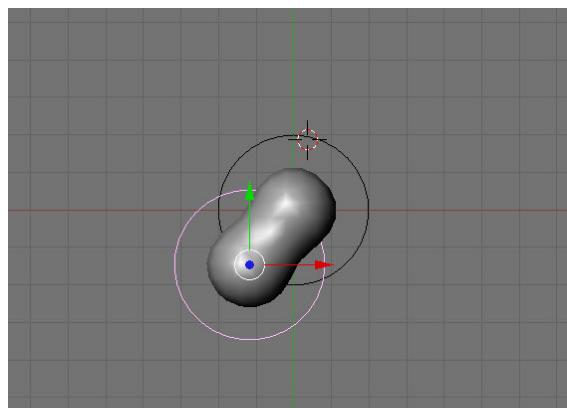
Right next to the Modifiers panel is the **Multires** panel. (Note: In recent versions of Blender this is not its own panel---it is a Modifier.) A multires object has the options to add numerous levels of smooth subdivision to a mesh. While you can use the sculpt tools on any polygon or nurbs mesh, the great strength of sculpting with Multires is the ability to jump back and forth to different levels, quickly sketching out the gross form at lower levels, and adding

finer detail at the higher levels. Add the Multires and Add a level; we're ready to sculpt!

Noob note: If nothing seems to be happening while you're trying to sculpt your mesh, it's because you haven't applied your Decimate. It must be locked down and applied before you're actually allowed to do anything to your mesh.

Noob note: Only apply multires once, doing it a few times will quickly increase the polygon count to amounts that will really slow Blender down

In case you haven't noticed, Blender's different modes offer different tool sets and options. To make the sculpting tools available you need to be in **Sculpt** mode, accessible through the Mode drop-down in any 3D window header. You'll see two new tabs next to the multires panel, *Sculpt* and *Brush*. Sculpt has most of the options you'll need to begin shaping your mesh. Note that most of the different brushes in this panel have hot-keys which will save a lot of time (**G**=Grab, **D**=Draw, **S**=Smooth, etc.). Most important here is changing the brush size and intensity. Pressing **F** and dragging the mouse will resize the brush, while **Shift-F** will allow you to adjust the Brush Intensity. You can also turn on **Symmetry** to paint, for instance, both sides of a face at one time. This can speed up tasks tremendously, as long as your mesh is aligned to the axis properly.



5.24 Match Moving

Match moving is the technique of recreating the position of the camera used in recording live action footage. This information can then be used within Blender to merge 3D objects with live action film. For a more detailed discussion of the concept, take a look at the [Wikipedia article on Match Moving](#).

Blender cannot perform match moving itself, you must use a 3rd party tool to determine the camera position and the way it moves, then import this data into Blender. While there are many software tools to do this, this page references two free options: Voodoo and Icarus.

5.24.1 Icarus

Icarus is a discontinued University of Manchester project which can be used for non-commercial work. The download links from the official page no longer function, but Windows and MacOS X versions are available from this [Icarus video tutorial by Colin Levy](#).

5.24.2 Voodoo

Voodoo is an actively developed free match mover available for Windows and Linux. Here is a tutorial on [using Voodoo and Blender](#)

5.25 Match Moving/Motion Tracking with Icarus and Blender

Motion tracking, also called **Match moving**, is an essential element when integrating 3D elements with live footage. Motion tracking software is usually pretty expensive, but the Icarus application (Windows and Mac) is available for free for educational use. Icarus, which hasn't been updated for while, was later replaced by the commercial application **PFTtrack**. Other popular motion tracking applications are **PFMatchit** and **PFHoe**(both also from The Pixel Farm), **Voodoo** (for Windows/Linux; free for non-commercial use), **SynthEyes**, **Boujou** and **3D-Equalizer** (commercial).

The excellent CG prodigy [Colin Levy](#) hosts Icarus(by kind permission of The Pixel Farm Ltd), the Icarus import script for Blender, as well as a splendid video tutorial (see [Download Icarus and Video Tutorial](#)). However, I lacked a brief text tutorial about motion tracking, so I decided to write my own. This tutorial is extremely brief and high-level, and requires some previous knowledge on video editing, 3D, and Blender.

Note: this tutorial was created using Mac OS X 10.5 Leopard, Blender 2.46, Icarus 2.09, and the [Icarus Import Script for Blender v1.07e](#) (for Blender 2.41, written

by Alfredo de Greef).

Tutorial

Phase 1: Preparing the Video Footage **Note:** this tutorial explains the *Auto-feature Tracking* mode in Icarus. There are other options which gives more user control - see the Icarus **UserGuide.pdf** for more information.

1. Record your video footage. Having the camera on a tripod (thus limiting to just panning/rotating) simplifies the tracking, but Icarus can handle a hand-held camera as well. Filming a background with orthogonal lines (that can align to X/Y/Z dimensions), such as a room, also helps the tracking.
2. Capture/import your video footage to your computer. Icarus handles video up to DV resolution (720*576 pixels).
3. Start the Icarus *Calibration* application (there is also a *Distortion* and a *Reconstruction* application).
4. Create a new project (Project->New).
5. Import your video footage (Project->Import Movie).
6. Fill in the *Camera Parameters* information in the window that pops up - especially the *Camera Motion* and the *Pixel Aspect* options.
7. In the left panel, expand the group called *Coordinate Frame*. You should see X Axis, Y Axis, etc.
8. Click the Z Axis tool (blue) and mark vertical lines in your video footage. Use the X Axis (red) and Y Axis (green) tools to mark horizontal lines (up to you to decide which should be X and Y).
9. Estimate the focal length (Camera->Estimate Focal Length).
10. Navigate in time in your video footage using the time slider (beneath the video image). Add more X/Y/Z marker lines on a few key frames, especially as new pieces of the background are revealed when the camera moves.
11. Save your project (Project->Save).
12. Start the tracking process (Camera->Track and Calibrate). This will take some time.
13. Export the results in human-readable form (Project->Export 3D Motion, select *Human Readable (*.txt)* as file type).

Phase 2: Importing the Motion Tracking Data into Blender

1. Start Blender, and open a Text Editor view.
2. Open the Icarus import script **ICARUS_import241.py** (File->Open).
3. Start the script (File->Run Python Script). You should now see the Icarus Import screen.
4. Press the *FSEL* button, and open the results you exported from Icarus.
5. Press the *Create Curves* button. This imports the camera motion from the Icarus data and applies it to the Blender default camera.
6. Press the *Feature Points Mesh* button. This imports 3D shape dots from the Icarus data, which helps as reference when you want to align your own 3D elements to the video footage.

You are now ready to add your own 3D elements to the Blender scene.

Phase 3: Compositing 3D Elements on top of Video Footage If you want to easily composite the 3D elements on top of an image, you can add the image as the rendering back buffer in Blender (Scene tab in the Buttons view). However, this doesn't work for videos, so we need another solution.

1. In Blender, switch to *SR:4 - Sequence* in the layout dropdown menu at the top of the screen.
2. In the *Video Sequence Editor* view (middle of screen), add your video file (Add->Movie). Move the new strip to layer 1, frame 1.
3. Add the current scene to the sequence (Add->Scene, Scene). Move the new strip to layer 2, frame 1.
4. Select the scene strip on the second layer (right-click).
5. Open the *Scene* panel in the Buttons view, and then open the *Sequencer* sub-panel.
6. Change the *Blend Mode* dropdown from *Replace* to *Alpha Over*. Your 3D elements should now render over the background video in the top-right preview screen.
7. In the Render panel, enable *Do Sequence* just below the ANIM button. This will enable the background video when rendering.

Troubleshooting

- If your imported Feature Points Mesh looks a bit spherical, you need to generate camera distortion data using the Icarus *Distortion* application.

5.26 Create a Clayman

5.26.1 Create a Clayman

(under construction)

This Tutorial was made using Blender version 2.41

Upon completing this tutorial, you will have made something like this:

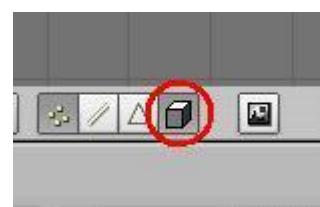


Completed Clayman Render

5.26.2 Making The Clayman

Upon start, delete the default cube (X-KEY) and switch to front view with NUM1. Now add a mesh plane and under The Edit Buttons Window, Mesh Tools Tab click the subdivide button. Now select the 3 vertices on the left and delete them. (B-KEY for Border Select Tool)

If you can't select the vertices then deselect the Limited Selection to Visible button.



Now select/deselect all vertices with A-KEY (we want them selected) In the Modifiers Tab in the Edit Buttons Window, Click *Add Modifier > Mirror* You should now see something like this.

Now select the 3 right vertices and extrude them on the X axis by movement of 1.0 units by pressing E-KEY (*Extrude > Only Edges*), X-KEY (to constrain it to the X axis) and holding CTRL to constrain it to simple movements. Do this twice.

Now extrude the legs by selecting the 3 bottom-right vertices and extruding them on the Z axis by 1 unit. Do this twice as well.

Now extrude the top of the clayman on the Z axis (in units

of 1.0) 7 times so that it looks like this.

Now let's make the arms by extruding the 2 boxes that are 3 boxes down from the top. Let's extrude it on the X axis by 1 unit 4 times so that it looks like this.

Now comes the extrusion of the whole body. As you can see, we only have a flat shape of a rough "Block". so lets move our view around with the NUMPAD arrows and extrude it on the Y axis. Select all vertices with A-KEY. *Extrude > Region* on the Y axis until it reaches 2.0 units. After done correctly, it should look like this.

With the Clayman successfully extruded, its now time to add some definition to him by adding a subsurface modifier. In the Edit Buttons window, under Modifiers, click *Add Modifier > Subsurf* and set the Levels bar to 2. It should look somewhat like this.

Now, Select all vertices with A-KEY, and click the Set Smooth button in the Links and Materials tab. Let's get to shaping the Clayman.Using the Border Select Tool (press B to get the Border Select Tool), grab the according vertices to make a shape like this.

after that's done, select the head vertices and grab them on the Z axis and move them up by 0.2 to make it look like this.

Now select all vertices with A-KEY, and scale them down (S-KEY) on the Y axis until it reads 0.6000. And finally, select the underside of the foot (the vertices that make it up) and subdivide it to make ir more flat on the bottom.

5.26.3 Adding Materials

Start by going to the Material Buttons window (F5) and Clicking on Add New, and going down to the column of 3 rectangles that have one gray one and two blank white ones. Click on the gray one and in the top-right corner of the pop-up window, you'll see a Hex Code. Click on the current hex code and type in this new hex code to get the right color: 73BCF7. Now hit F6 to go to the Texture buttons window and click on Add New. Change the Texture type to Stucci. Now you'll see a new tab appear. On it, for NoiseSize enter 0.150. Now go back to the Materials Buttons window. On the far right there will be three tabs in one. On the Map To tab, Deselect Col and select Nor. Now your example should look like this.

5.26.4 Inserting Armatures

Now lets move on to inserting armatures in our clayman. Lets start by going into object mode (alternate between Object and Edit mode with TAB) and adding an armature. Press *Space > Add > Armature*. It automatically puts you in Edit mode upon adding the object. Switch to Object mode and grab it along the Y axis by 1.0 units so that it's inside the clayman. Sure, you can't see it now, but turn on X-axis Mirror Edit and X-Ray in the Edit Buttons

window.

Now switch back to edit mode, and select the top of the bone. Etrude it on the Z axis by 1.4 units two times so that the tip of the second extrude is right in the center of the two arms. Now extrude it on the X axis by 3.0 units, then grab the tip of it and move it on the Z axis by 0.1700 units. Now extrude it on the X axis by 2.0 units twice. And do the same to the other side. It should look like this.

Now let's extrude the neck. Extrude the top bone along the Z axis by 1.0 units to make a pretty good size neck. Then extrude it up on one side so that the position reads like so. If you can't get it exact then at least try to get it as closest to the number as you can.

Then do the same to the other side so that it looks like 2 antennae. Now let's extrude the hips and legs. Lets start by extruding the bottom bone by 2.0 units on the X axis and for the Z axis, -1.0 units. Then extrude the foot by -2.0 on the Z axis. And last, select a forearm bone (select a whole bone by clicking the middle of it) and in the Edit Buttons window, you'll see a tab called Armature Bones. Deselect the Con button in that tab, and do the same with the other forearm bone. This allows us to be able to move the forearm out, instead of having it locked to another bone.

5.26.5 Applying The Armature

Ok, now we're going to apply the armature to the mesh so that when we move the bones, the mesh moves along with them. Lets start by switching to object mode and selecting the clayman mesh, and under the Modifiers tab, click apply on the Mirror modifier. You may see a pop-up, just click OK. Not only have we just made room in the Modifiers tab, we applied the mirror modifier to the mesh, so now we will have to edit both sides if we want to change something. But worry not, we wont need to change anything from here on. With that new space in the modifiers tab, let's add an Armature modifier. You may see a line that says "OB:" type in the name of the Armature in here (the default name for an armature is "Armature" unless you have made more than one) also, deselect the Envelopes button,for we will not be using envelopes in this tutorial. Here is what you should have.

Now, select the Armature and press CTRL + TAB. If you already have a bone selected, it should be highlighted blue now. We have just switched to Pose Mode. Now, while in Pose Mode, select the clayman mesh (if you cant select it, zoom in closer to get some of the bones out of the way) and press CTRL + TAB on the clayman mesh. We have just entered Weight Paint Mode. This allows us to assign vertices to the bones so that the mesh moves with the bones. Let's start by setting up the weight painting. You should see a new tab called Paint in your Edit Buttons window. Adjust your settings so that they conform to this.

Start by selecting the top-right bone of the head and start weight painting the front and back (you'll have to switch your view so that you can see the back to get the painting in there) Weight Paint it according to this.

Now Weight Paint the Neck. (pictures not shown of back view, but be sure to get the back anyway)

Now the Shoulder.

Now the Forearm.

Now The Hand.

And after getting the other side of the body, let's get the upper chest.

Now the middle chest.

Now the lower chest.

Now for the hips.

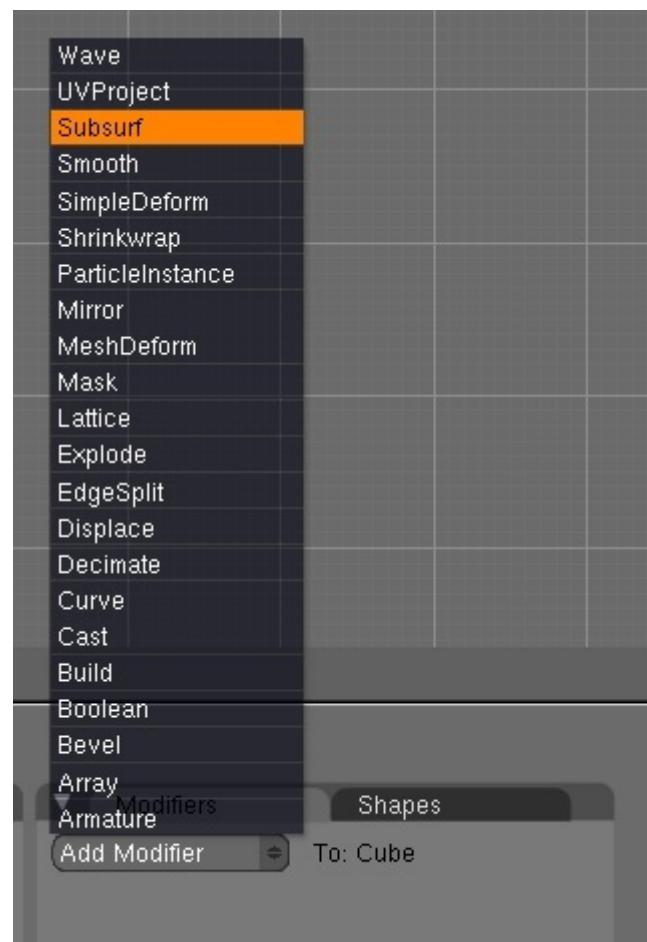
Now the Feet.

After doing the other side of the leg, switch back to Object Mode with CTRL + TAB, and select the Armature. You should already be in Pose Mode with the Armature. Now select a bone and grab it to move it. if you have Weight painted correctly, the Clayman mesh should move with the bone. There's two tips to moving the bones around in Pose Mode.

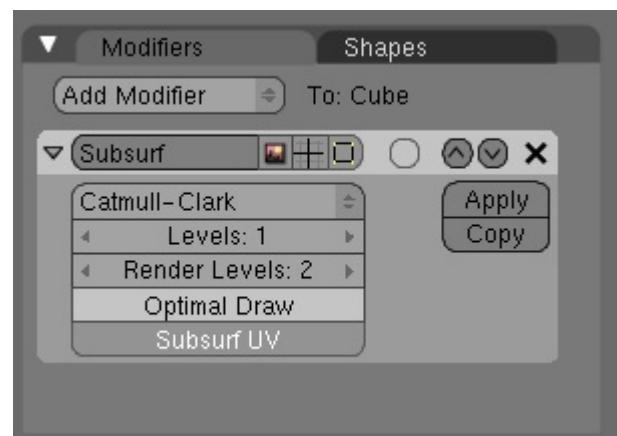
1) In the Edit Buttons window, whether Automatic IK is turned on or not affects how the bones move.

2) Where your view is affects how the bones rotate. (unless you have it constrained to an axis)

If you find that a part of your clayman isn't moving, just switch back to Weight Paint mode for that bone and repaint it.



In Blender 2.5 Alpha 0, Subsurf has been removed from the modifiers list. Instead hit Ctrl, followed by a number which will specify the subsurf level. For example **Ctrl-1** adds a new Subsurf modifier of level 1. This also works in Blender 2.4. Increasing the level greatly increases the number of verts in your model, so make the level relative to the number of vertices in your original model (pre-subsurfed). Subsurf often works best in conjunction with smoothing, so be sure to set your object to smooth, again in the 'Editing' tab, or in 2.5 A0 under 'Object Tools', which you can bring up by hitting the T key in the 3D viewport.



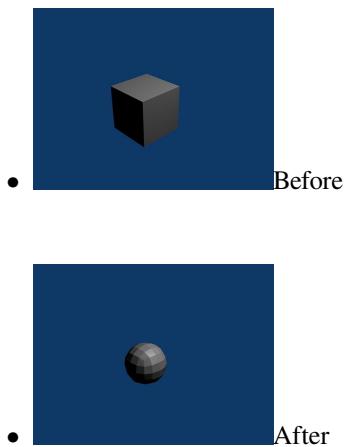
In this way you can create a smoother, higher-poly shape

5.27 Organic Modeling

Subsurface modeling

Organic modeling is considered by some the most challenging. Non-organic can mostly be accomplished by extrusion and scaling. Organic modeling on the other hand involves mainly curves, as nature has a thing against straight lines. Because of this, organic modeling is usually done with subsurfaces. To subsurf a mesh, first select it and navigate to the 'Editing' tab. Then go to the modifiers stack and add a new Subsurf modifier.

based on that of the original mesh. This is controlled by the vertices, edges and faces of the original. In order to control the rounding of your mesh you can use two methods: loopcuts, which is a very sloppy method as it adds more verts to your mesh, which serve very little purpose and can get in the way of modeling; and Edge Creasing. The latter can work extremely well in most circumstances. By default all edges are declared uncreased, and so allow complete rounding in subsurfing, by creasing the edges using **Shift-E** you can far better control the amount of rounding on your mesh, and most importantly, without adding extra vertices. However, unfortunately Edge Creasing is not available in Blender 2.5 A0. Yet. And so loopcuts and extra verts are the only option.



Now, start shaping your mesh into an organic shape. There are various tutorials for modeling bodies and faces, but it is often a good idea to use references. Only use quads - that is, shapes with four vertices. Triangles do not subsurface well, they create bad looking rough surfaces - avoid if at all possible. Also avoid Poles, which are vertices connected to anything other than 2 or 4 other vertices, as they create literal creasing in your subsurfed mesh, that is, sudden bunching up and pinching of faces.

To make the object shape properly, you will need good edgeloops and edgeflow. This means that all your vertices and/or faces line up in a continual line or curve around your model. It is possible to select edgeloops all in one, fantastic when you have to resort to them in controlling rounding, by holding down **Alt** when selecting a vert. This really expands into topology, which is an advanced subject. I would recommend visiting <http://www.subdivisionmodeling.com/forums>, although it has just been closed down, which was a great place to start, however it still (at the time of revision) is active and well worth a look if you are struggling with modeling heads. It is best not to have too many vertices to avoid making the subsurfaced shape look rough - in other words, the original shape should be quite low-poly.

5.28 Understanding the Fluid Simulator

5.28.1 Understanding the Fluid Simulator

When I first encountered the Fluid Simulator I had a hard time understanding its behavior, especially the *Start time* and *End time* didn't seem to make any sense at all. Going on a Google spree revealed that many people have problems figuring out the secrets behind fluid simulation, and I didn't find any truly helpful guides. In this little guide I'll try to explain it in a newbie friendly way. It may not be entirely correct, although it might help newbies understanding how it works.

5.28.2 First of all

Start time and *End time* are in seconds. Don't forget this. Even if your simulation seems to go insanely fast when you set *Start time* to 0 and *End time* to 10 and having 250 frames in your animation with 25 frames per second, there is a good reason for this. For now, just remember this, don't let your mind wander and believe that the values are in milliseconds or that you have to do some wicked math dividing/multiplying with frames and so on.

Also notice that the domain is the bounding box for the whole fluid calculation. EVERYTHING is done inside this box. It acts as the floor, ceiling and walls for all of the fluid. This is very important, as the number 1 reason why I couldn't get a good fluid simulation going. If I have time, I will do a section on Fluid > Control, but for now, I will say that it adds a LOT of calculation time. Running on an I7, ATI Radeon 8970 Video, Asus P6X58D Premium, 64-bit Windows 7 and 64-bit Blender, I crashed my computer with the lowest quality settings. So just remember that the domain MUST surround the area in which you do calculations. Also note that after you set up your simulation, the domain becomes the actual liquid, so give it proper material and try not to bake until you're pretty sure of your simulation.

5.28.3 Setting up the scene

We'll learn how the fluid works the practical way. plane

- Start with the default box, this simulation will be very simple.
- Let's work in wireframe mode, press the **ZKEY** to turn off solid mode.
- Go in to camera view by pressing **Numpad0**.
- With the box selected, scale it up to two times by pressing **SKEY**, then **2**, then **ENTER**. This fits the camera fairly well.

- Press **Numpad7** for a top view.
- With the box selected, press **Shift-D** (don't move the mouse or else the duplicated box will move) then press **ENTER** to confirm the duplicated box's location. If you do move the box, just press **Escape** then the new box will be kept but the move cancelled.
- While the new box is selected (and in exactly the same spot as the other box), press **SKEY**, then **.5**, then **ENTER** to scale it to half size.
- Stay away from the mouse, accuracy is important here and I'll explain why later.
- Now we want to move the new box into one of the upper corners. Press the **GKEY**, then press the **XKEY**, type in **-1**, press **ENTER** and the box should move to the left wall of our larger box.
- We want the box in a corner, so press **GKEY Y 1 Enter**, the box should now be in the top left corner from our current view.
- However, we're in 3 dimensions, not 2 so click **Numpad1** for a side view. This time we'll move the box up along the Z-axis: **G Z 1 Enter**.
- Excellent, our setup is done.

5.28.4 Setting up the simulation

- Make sure you're in Object Mode, and that you followed the above steps precisely.
- Select the smaller box and click **F7** twice. You should get a panel where the rightmost pane says "Fluid Simulation". Click **Enable**.
- Our small box will be the fluid, so just click the **Fluid** button. That's all there is to do with the small box.
- Now select the large box. The Physics panel should still be visible, click **Enable** in the Fluid Simulation pane and then select **Domain**.
- By default your animation should have 250 frames. Rendering should also be set to 25 frames per second by default, this tutorial assumes this setup.
- Since we got 250 frames and 25 frames per second that means our animation is 10 seconds long. So here comes the tricky part, which actually isn't that tricky at all:
 - Start time is by default set to 0 seconds. This means that on the very first frame the simulation has just begun. You could increase this value to say, 1 second and that would mean that on the first frame the fluid simulation has already run for 1 second. We don't want to do this, so keep it at 0 seconds.
 - End time is by default set to 0.3 seconds. What does this mean? This means that on the 250th frame the simulation has run for 0.3 seconds. However, by default our animation is 250 frames long with 25 frames per second, making those 0.3 seconds stretched over 10 seconds. Basically this means that we're watching the show in slow motion, or slightly less than 1/33 the realtime speed. So now you may think "it looks quite realtime to me!", and yes, it does, but why does it do that? Well, that's hard to explain. Consider this: In a world without friction, how far would a drop of water fall after 1 second? The answer is about 4.9 meters. So, if a drop of water falls from 4.9 meters it will take 1 second before it reaches the ground. How long would it take the waterdrop to reach the ground if it falls from 3 centimeters? About 0.078 seconds. So why do I mention 3 centimeters? Because by default the size of our domain is 3x3x3 centimeters, or **really** small. If you're like me, you were probably thinking that the fluid was flowing around in a bathtub or a barrel, not in the wrapping of a cupcake. Set **End time** to **10 seconds**.
- Since our imagination likes big things, let's crank up that cupcake to say, a swimming pool. Make sure the big box is selected in Object Mode In and look at the Fluid Simulation pane. Just to the left of the "BAKE" button there should be 3 other buttons, possibly named "St", "A", "B". Click **A** for advanced options.
- Some new boxes should appear, Gravity (should be **-9.81** for the Z-axis, nothing else), Water and the option we're looking for, "Realworld-size". Also Gridlevels and Compressibility, but let's not care about those now.
- The "Realworld-size" value says how large our domain is in meters, and as you can see it's 0.03 meters by default, or 3 centimeters. We want it huge, so crank it up to 10, which is the limit for Blender 2.45. Now our swimming pool is 10x10x10 meters (don't drown!), remember this because scale matters with fluid. Do not think we're playing with cupcakes again :)
- Now click **BAKE**, and read on while your computer is chewing zeroes and ones.
- Remember how I told you to be very accurate about placing that second box? And how I began talking about gravity, falling waterdrops and stuff? Well, now you're going to see why.
- As stated, our "swimming pool" is 10x10x10 meters. The smaller box we added is exactly half the size (well, in terms of length/width/height, not volume), or 5x5x5 meters. Remember that a drop

of water would fall 4.9 meters in 1 second? And that our animation got 25 frames per second? This means that the bottommost part of our blob of water will be exactly 5 meters above the “ground”, and that after 25 frames our water should be very close to the ground.

- If you got a fast computer, Blender should be done baking by now. Go to frame 25, for example by using the arrow keys (up/down goes 10 frames forwards/backwards, right/left goes 1 frame forward/backward). Take a close look at the blob, then go forward 1 frame. Notice how the blob hits the ground? Rings a bell, doesn't it? :)
- Although, we're not done! We gotta render our swimming pool. It's easy, but takes time, hit **Ctrl-F12** and go make dinner.
- When the rendering is done, press **Ctrl-F11**, and think of a 10x10 meter large pool. You might want to keep an eye on your kids if your local swimming pool acts this way, though.

5.28.5 Final notes

Scale matters. It's really difficult to understand fluid dynamics on a very small scale, especially when you don't even know what scale is used. The “Realworld-size” value seems to be left out in many guides, I would recommend you set it to something you can relate to, or you'll end up with simulations that look really slow/fast or having an **End time** value that seemingly makes no sense. Further I'm not a mathematical genius, for all I know I could be way off with my explanation, although this way the values makes sense to me, and I'm able to make fluid simulations without “guessing” on values for **End time**.

5.28.6 Extra Practice

This YouTube tutorial on fluids might also help: [Link](#) and this [Realistic Water Texture](#)

5.28.7 Links

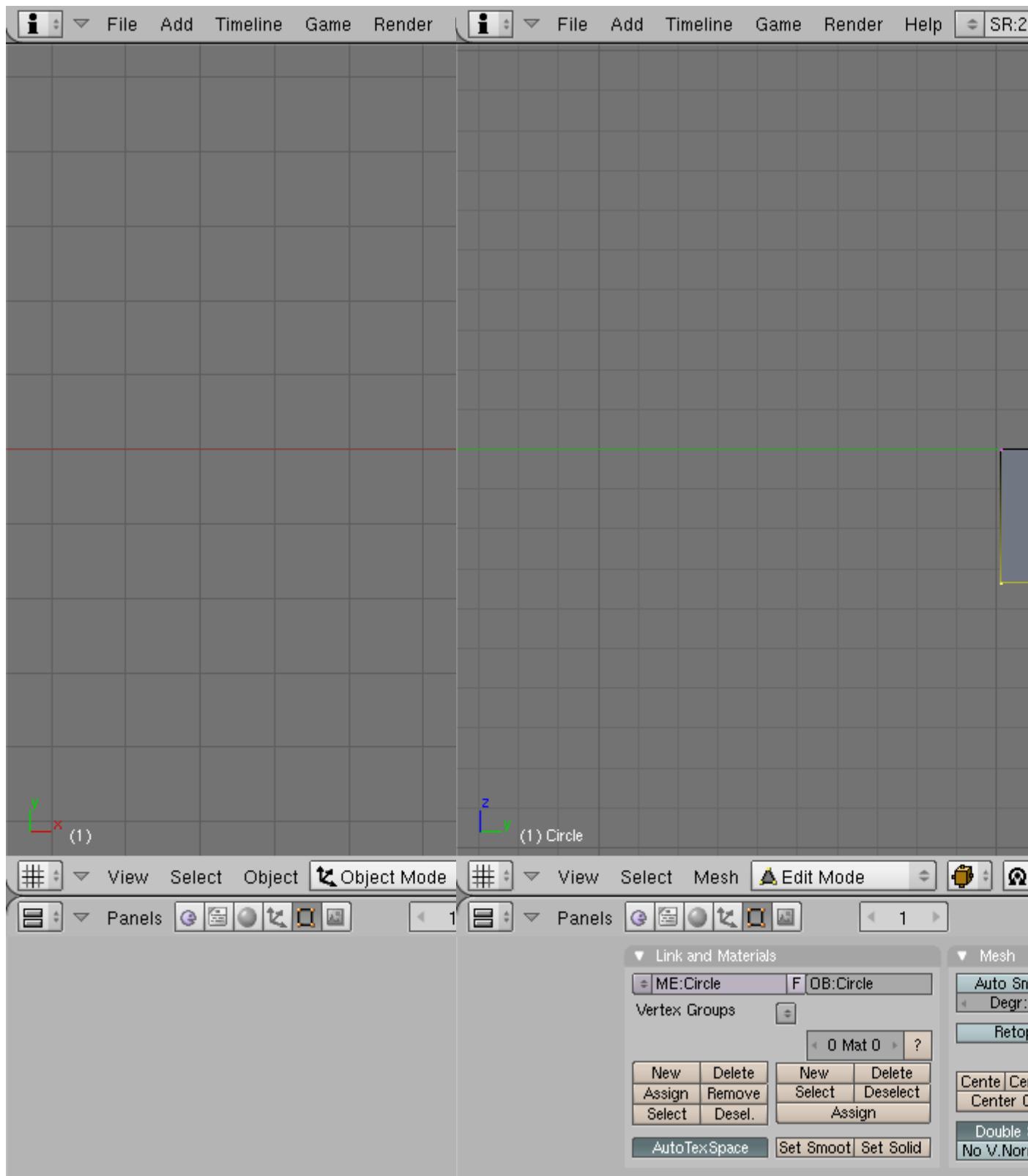
- Youtube: Comparison of different values for resolution and real-world size

5.29 Creating a jewel in Blender

In this tutorial we will create a jewel in Blender. It is fairly simple. I recommend you do this tutorial if you are a noob, because it explains some basic features, but I suggest you read the tutorial syntax and the pages at the very beginning of this Wikibook first.

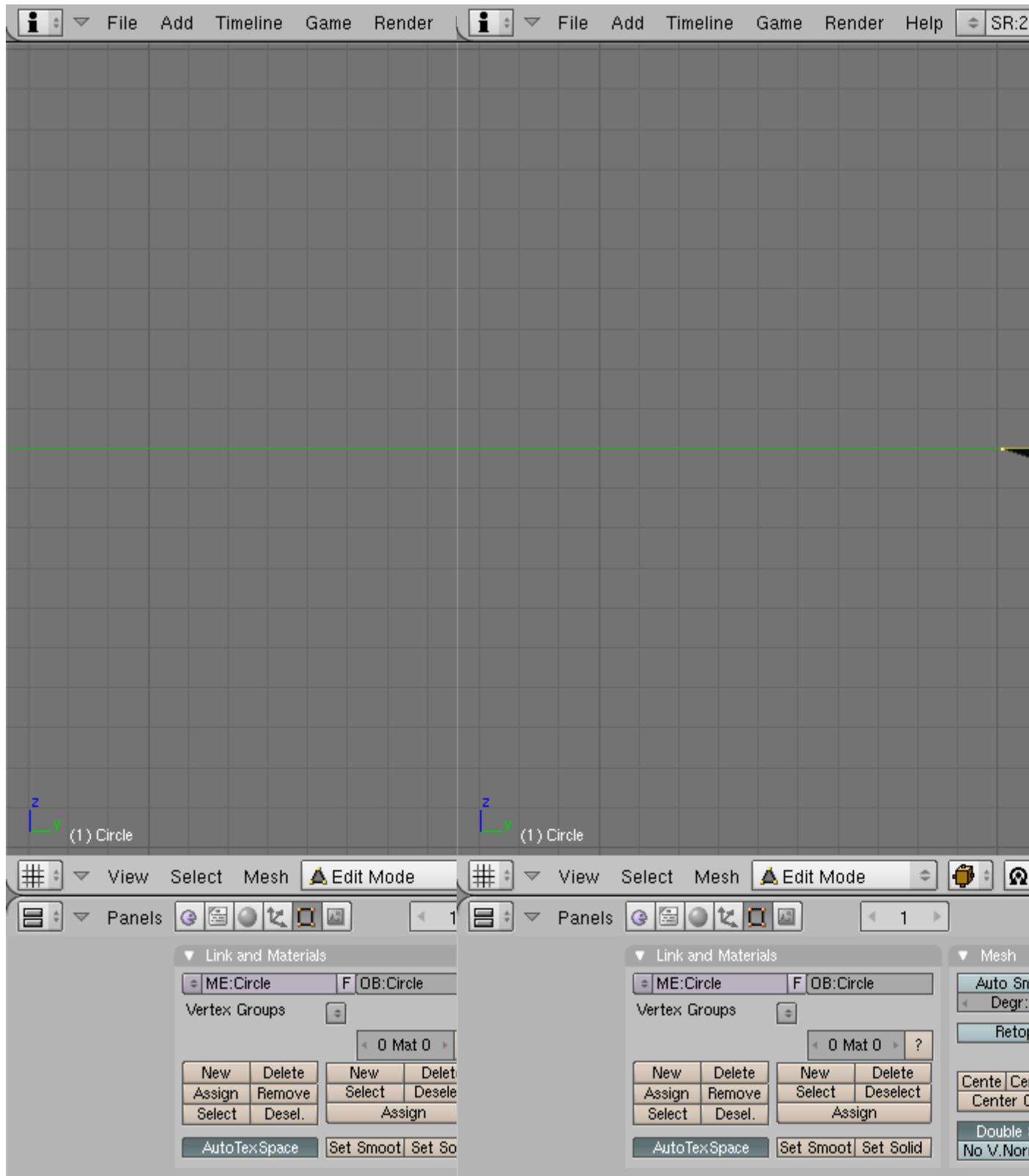
5.29.1 Modeling the jewel

Start up Blender if you haven't already. There is a cube in top view (it looks like a square because it is in top view.) Delete it by pressing the X key. Then, to begin modeling, add a circle by pressing **SPACE**→**Add**→**Mesh**→**Circle**. Set the vertices to 10 and make sure it is not filled in. Then press OK.



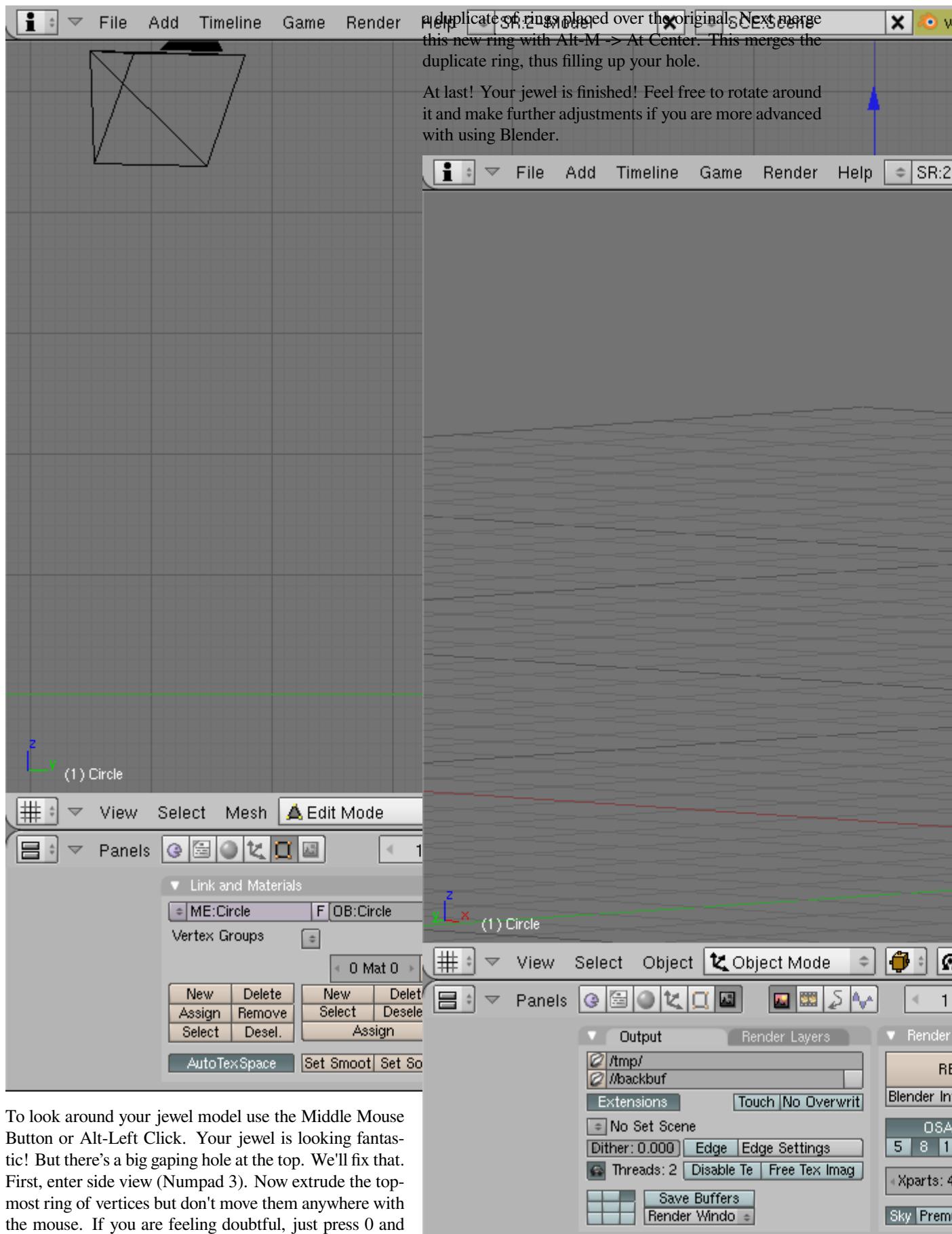
Next, we will extrude the vertices down. First enter side view by pressing Numpad 3 on your keyboard. As you can see, the circle is flat and not filled in. We will give it some depth. Switch to Edit Mode (TAB) and extrude the circle by pressing the E key and selecting “Only Edges”. Move the mouse down and click to confirm the position (you may want to limit to Z axis by pressing the Z key).

OK, what we just did is turn the circle into a hollow cylinder by extruding. But we don't want a cylinder. We want the bottom to be a nice tip. To do this, press Alt-M. This creates a Merging menu. Select “At Center”. Now the bottom is a nice tip, like we want it!



Now we will edit the top of the jewel. Press A to deselect everything and press the B key. This enters Box-Select Mode. Drag a box over the top vertices to select them.

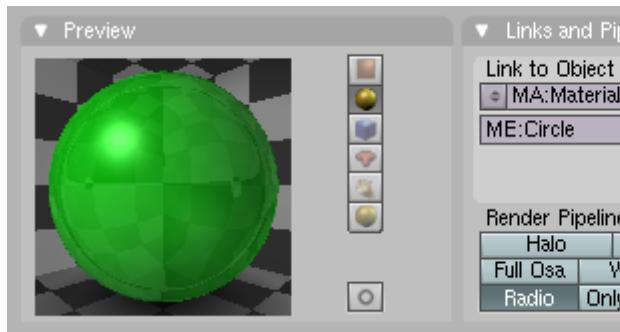
Now extrude the vertices upwards a bit (again: E key and “Only Edges”). Next, we will scale down these vertices to look a bit more like a jewel. Make sure the ring of vertices are selected and press the S key. This enters Scale Mode. Scale down the top vertices a bit and click to confirm.



To look around your jewel model use the Middle Mouse Button or Alt-Left Click. Your jewel is looking fantastic! But there's a big gaping hole at the top. We'll fix that. First, enter side view (Numpad 3). Now extrude the top-most ring of vertices but don't move them anywhere with the mouse. If you are feeling doubtful, just press 0 and then Enter (or just click right mouse button). This makes

5.29.2 Shading

Switch back to Object Mode (TAB) and show the Shading Panel (F5). Enable the Material Buttons (click on the button with a red ball). Add new material by clicking on “Add New” button in the “Links and Pipeline” tab. Copy the settings in the screenshot below. If you are having trouble setting the precise value to a slider, just click on the number. This allows a manual type-in. First, make it green by setting R to 0, G to 1, and B to 0. Next set the alpha slider to 0.458 (it looks like an A). Now press “Ray Mirror” (raytracing reflection) to turn it on and set it to 0.13. Now press “Ray Transparency” and set the IOR value to set 1.44. When you’re done it should look like this:



5.29.3 The finishing touch

To improve the effect, add a plane underneath the jewel. In the Object Mode (TAB) press SPACE>>ADD>>MESH>>PLANE. Scale it 5 times using the S key. Then move it down a bit using the G key (press the Z key to restrict movement to the Z axis).

You may try to render your jewel now. Press the F12 key. You may find that camera doesn't see whole jewel. Move and rotate the camera (using the G and R keys) to set it in the right position. You may want to switch to “Camera View” (Numpad 0) and try out “Camera Fly Mode” (SHIFT F). Try also moving the lamp and see what happens.

I've found that the jewel looks best with Ambient Occlusion on. So go to the Shading window, then the World buttons, click the Amb Occ tab and click the Ambient Occlusion button.

Here is an example of what you can do:

5.29.4 External links

- <http://www.blender.org>
- <http://www.blenderartists.org>

5.30 Modeling a picture

5.30.1 Modeling a picture

Ever seen an awesome looking picture you wanted to turn into a 3D model? Like a logo or a symbol? Well, it's actually pretty easy... it just takes some time to do.

- First off, you're going to need a picture to trace. I'm currently doing a project for a friend to do with devils and demons, so I chose a demonic looking face for this tutorial: [Demonic Face](#)

- Now open Blender and start a new project. Delete the default cube. Before you start tracing the face, you need to set the face as the background image. To do this, click 'view', then 'Background Image'. A box should pop up with only one button in it (Use Background Image), click it. Now some settings appear, we're only interested in one of them for this tutorial. Click the small button with a picture of a miniature folder on it (it looks kind of like a feather pen). It's the first one under the **Use Background Image** button. From there, select the picture you want to trace. Like this: **Background Selection**
- OK, now for the long part. Zoom in to the new background image just a little bit. Now, add a Bézier curve, and size it down a little. Hit **F9** and, in **Curve Tools**, find and click the **Poly** button. Now there should be a few more vertices to work with and the curve should be just a bunch of joined lines. Select one point at a time and using the **GKEY** move it to a point along the background image(or face in this case). Do the same for all of the rest of the vertices, making sure you only have one vertex selected at a time or you'll move more than just the vertex you want to. Once this is done, select one of the end vertices of the curve (it doesn't matter which end) and use **SHIFT+DKEY** to copy that vertex. Move the newly copied vertex to a point along the edge of the face a small ways away from the vertex you copied it from. Continue doing this until you have a complete outline (of the whole face or just one part, like the ear). Here's what it should look like (I did the left ear): **Tracing**. You can't see it in the picture, but six of the points on the right side of the ear are connected, while the rest aren't. In order to get the effect we're looking for here, we need to connect all of the points around the edge to make an outline (make sure not to connect the points across the picture or you'll have a messed up outline).

(user note: hitting **CTRL-LMB** instead of **SHIFT-DKEY** will add a vertex that is already connected.)

- To get the outline for the whole face, just do exactly the same thing around all of the edges. We still have a problem though: most of the points aren't joined by a line, so all we have is a bunch of dots. This is easily solvable. Using the **BKEY** or the right click of the mouse, we select a bunch of vertices at a time (somewhere between 5 and 10), and hit the **FKEY** a few times. Every time you hit the **FKEY** it should connect two of the points. Do this until all of the selected points are connected, then deselect them and select another group and use **FKEY** to join them. Keep doing this until all of your points are connected. To connect the last two points, select all the points and press the **CKEY**, to close the polygon.

[edit: A better option would be to select a vertex on one of the ends of the whole line, hold down the **CTRL** and left-click on a certain point on the image. This will create a new vertex, immediately connected to the vertex you selected.]

- Now that we've got the entire face traced (or outlined if you want to call it that), we can make it 3D. Hit **F9** again and find the Ext1 and Ext2 properties, shown here: **Ext1 & Ext2**. Change the values and see what happens. They correspond to the depth of the outline. Try changing them around until you find what looks good. Now, you'll notice that the lines just stick out straight. I'm still investigating how to actually model a head from the outlined face ... so if anyone has any ideas, feel free to add them to this page.
- In order to make it have depth you should make the outline out of mesh points instead of a curve. Add a primitive mesh and delete all the vertices in edit mode, then ctrl click to all point outline. Add depth to the surface in a side view (split views so you can see what you're moving). It helps to have 2 or more reference images, but you can wing it. Usually the final result has to be subsurfed.

(USER EDIT: I accidentally started it with mesh instead of curve. You can do the same thing with extrude, but I have no idea how to go on after that) (USER EDIT LATER: If you subsurf it, it creates a relatively 3D looking image. Its really cool)

(Another user, even later: If you want to turn your curve into a mesh, hit Alt-C while in Object Mode. Note that this is NOT reversible.)

(user edit: You can delete one vertex of a plane, in order to get a line. You may find easier to outline the picture extruding and moving points of the line you created.)

(user edit: you can use this tracing technique to make solid and symmetric models, else, you would really have to use normal modelling)

5.30.2 Printing a Rendered Image

Render your image. Exit or minimize the "Blender: Render" window. In blender, go to File -> Save Image... Then save your image. Then you can print it as you would print any other picture, using The Gimp, Paint, Microsoft's Photo Editor, or many others.

5.31 Modeling with the Spin Tool

The Spin tool is a great tool for modeling objects you might make on a lathe quickly and easily.



A collection of objects modeled with the spin tool

Lathe objects have circular cross sections along a certain axis. That is to say, when you cut such objects perpendicular to a certain axis you'll get circles. Examples of such objects includes rods, poles, wineglasses, and pails. A Blender render at the left shows some of these.

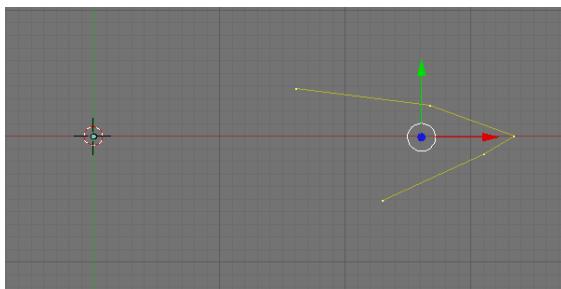
With the Spin tool you only have to “model” half the outline of your object. The object is completed after you spin this outline. A bit of cleanup here and there and your model is finished.

5.32 Spin Tool Introduction

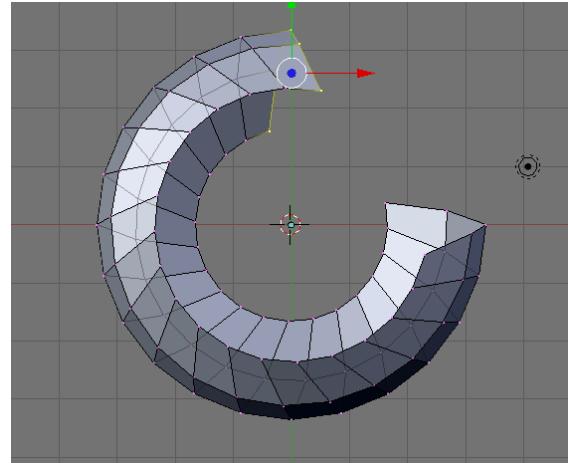
5.32.1 How the Spin Tool Works

The Spin tool works by making copies of the selected vertices in a radial array around an axis indicated by the 3d cursor. Each copies of vertices are connected to previous vertices that corresponds to it with necessary geometry, i.e. edges and planes.

A picture explains this more clearly.



Selected vertices before using the Spin tool



Result after using the Spin tool

Note that the copied vertices are arranged in a radial array around our 3d cursor and parallel to our view plane.

5.32.2 Typical Work Flow

- 1) Make half of the outline of our object with connected vertices. If you want to model a wineglass, make the half of the outline of the wineglass.
- 2) Spin our outline. After this the Spin tool would complete our object.
- 3) Clean up the resulting mesh. The mesh created by the Spin tool is not so perfect so we had to “clean” it up.

Modeling Half of the Outline To start we make half of the outline. Here we do it by creating vertices.

Open Blender. Don't remove the default cube since we are going to use it. If you don't have any object after you open Blender (like mine), add a mesh object. Any of them will do. I tend to use the plane. At this time it is good to give your object a proper name. Name it sensibly, like “wineglass” if we are modeling a wineglass. This will help us locate it if there comes a time when we have a lot of objects in our scene. We don't want to have all objects to be called cube.001, sphere.003 or something. That could get confusing.

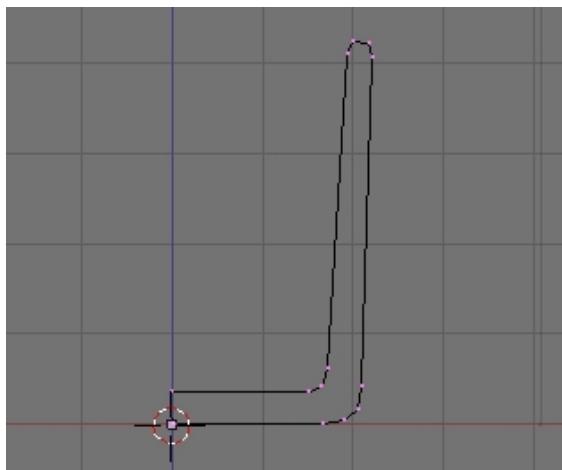
Hit Tab key to enter Edit Mode, if we are not in there yet. Select all vertices by hitting AKey, then delete them with XKey -> Vertices. Then go to the front view by hitting Num1.

Now, we are going to make the outline by creating vertices. Press and hold CtrlKey then click the left mouse button. A new vertex will be created. Click the left mouse button again, another vertex will be created but this time it is connected to the previous vertex by an edge. Click again to add more vertices. We are going to use this to create our outline.

You might notice that the 3d manipulator (red, green and

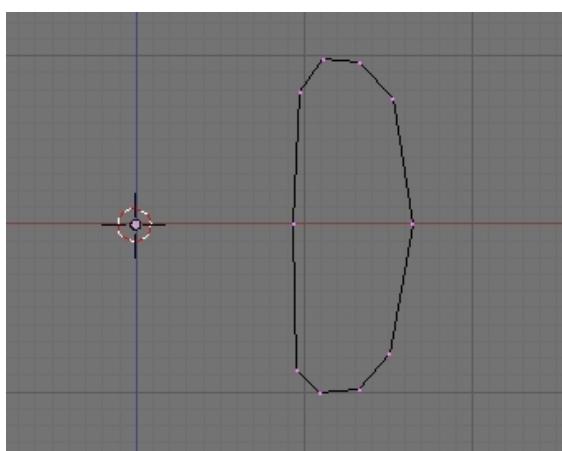
blue arrows) could get in the way. Deactivate it by hitting Ctrl+Space Bar, then select Enable/Disable, or by clicking the white pointing hand icon in the 3d view window.

Let's start making the outline. Delete our previously created vertices first. Using what we have learned, let's create vertices again but this time let's add them to form some sort of outline. Let's be creative here. It could be anything you want. Here I've made two to illustrate two situations that we might encounter when modeling with the Spin tool. That is, modeling objects that are hollow or non-hollow.



Half outline of a drinking glass

The first one is an outline of a glass. Here, we are modeling an object that is not hollow. We had two vertices that are located in the center of our would-be glass bottom. We need to align the two central vertices perfectly along the z axis. Select this two vertices then scale it along the x axis to zero (SKey, xKey, 0Key, Enter).



Half outline of a napkin ring

The second one we will make into a napkin ring. This time our outline is closed and no vertices are located in the center of our would be napkin ring. Here we are modeling an object that is hollow.

To make the closed outline, select the two vertices on

the open side then press FKey. The two vertices will be bridged by an edge, thus closing the outline.

Adjust vertices when necessary.

For those who may like to have an image in the background to guide them. Upload a background image by selecting the View menu from the 3d View Window then select "Background Image". A new internal window will appear. Click "Use Background Image" Button, then click "Load". Navigate to where the Image is located, select it, then click "SELECT IMAGE". Our image will appear on the 3d Window as a background. Below, I used a picture of a wineglass.



Half outline of a wineglass with a background image as a guide

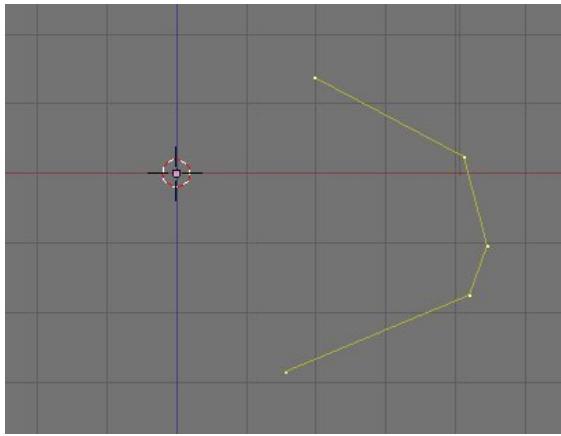
Spinning Our Outline Now that the outline is finished, we are ready to use it with the Spin tool

Two things you must know is that the Spin tool is dependent on the location of the 3d cursor, and view where we activate our Spin tool.

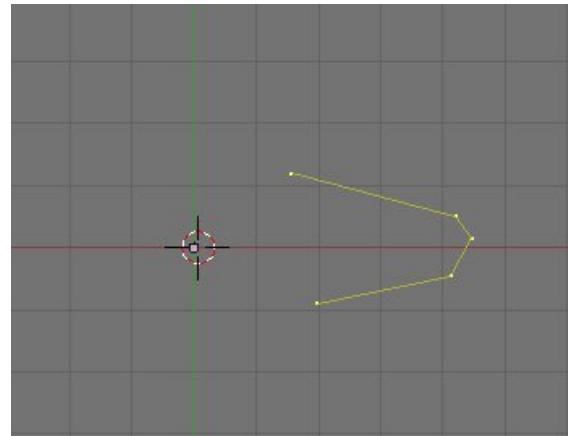
Here is our setup with the vertices all selected. In the next pictures are results when this shape is spun with the 3d cursor at different locations.

Here we have the same shape but is spun around in different views.

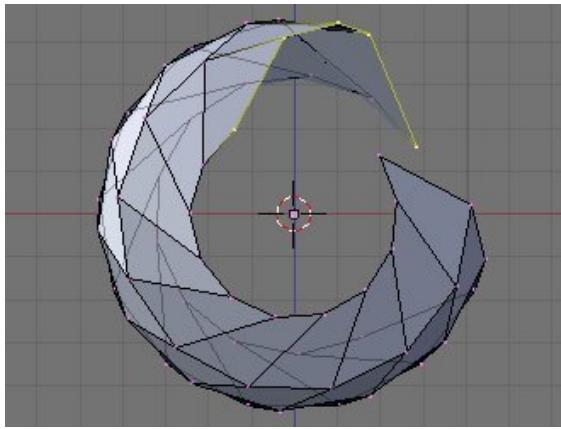
Note that its important only to place the 3d cursor properly relative to the view where you are going to use the



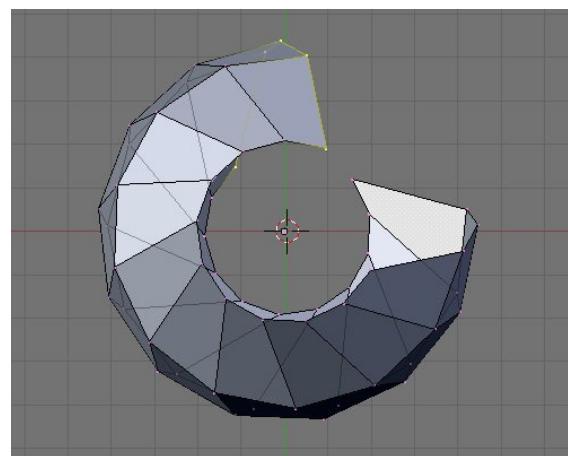
Setup with an arbitrary shape



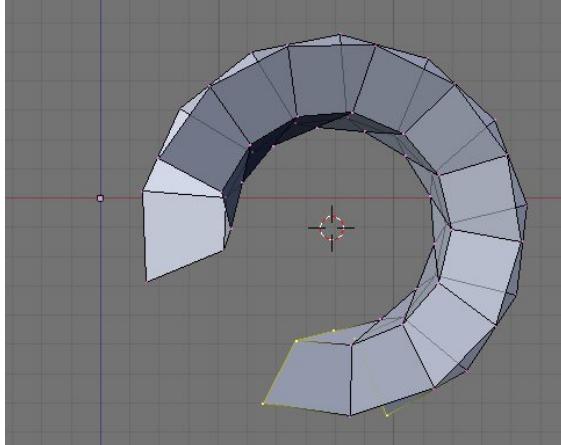
Shape seen in top view



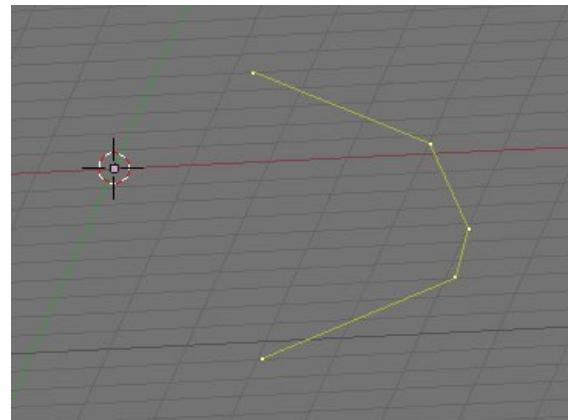
Shape spun with 3d cursor at the origin



Shape spun in top view



Shape spun with 3d cursor at another location



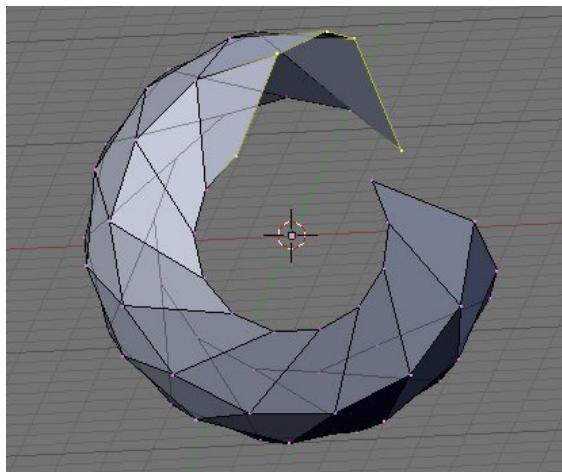
Shape seen in an arbitrary view

Spin tool

Placing Our 3d Cursor We can see that the location of our 3d cursor is very important in Spin modeling so we need to know how to place it properly. If you prepare it to be, the cursor would already be in position after we make our outline but accidents happen and we accidentally misplaced our 3d cursor. Here are some ways we could make our 3d cursor go to the place where we want it to be.

tally misplaced our 3d cursor. Here are some ways we could make our 3d cursor go to the place where we want it to be.

Snapping Snapping the 3D cursor is a quick and simple way to place our 3d cursor to its proper location. In the 3d window, hit Shift + SKey. A list of options would appear.



Shape spun in an arbitrary view

We are interested with the last three options, namely the cursor snaps options. They are the Cursor->Selection, Cursor->Grid and Cursor->Active. The names are pretty descriptive but we'll give a short description here.

Cursor->Selection places the cursor to the exact location of our selected element or elements. In case of multiple selected elements, our cursor is placed on the median of the selected elements. In edit mode, select a vertex and use Cursor->Selection snap to it. Our 3d cursor would jump to the exact place where our selected vertex is located.

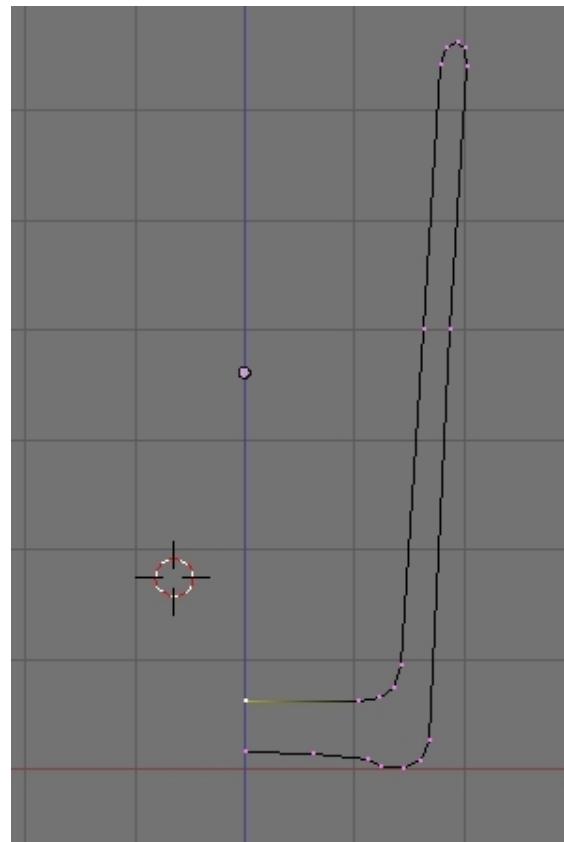
Cursor->Selection snapping is especially useful when modeling non hollow objects because there are vertex or vertices located at the center of our model to snap to. Just select one central vertex and snap to it.

Cursor->Grid snaps the cursor to the nearest cross of the visible grid. Try clicking at a center of a square in our Blender's grid and use it. See its effect.

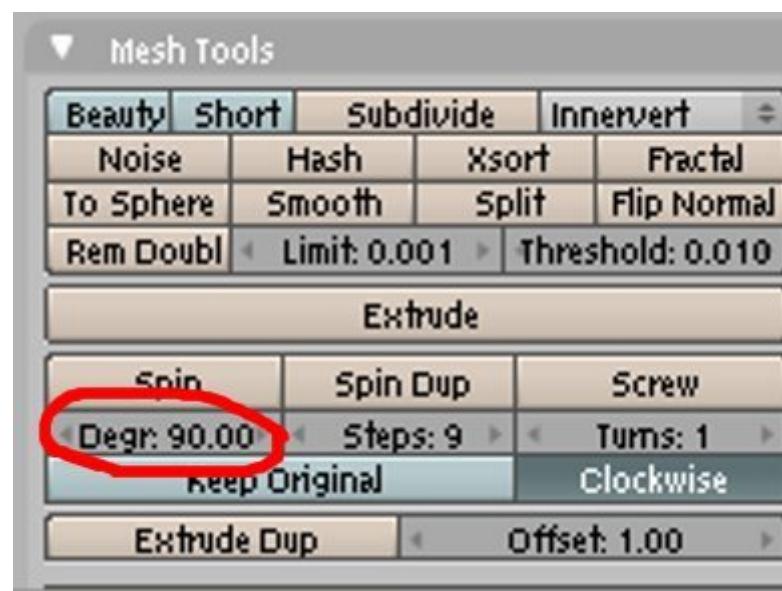
Cursor->Grid snapping is useful for spin modeling hollow models. Usually if you are careful, our object outline would be on a plane that contains one of the axes. If we make our outline in front view as advised, go to the top view (or side view depending on what we are modeling). If you are doing okay, you will see all our vertices aligned perfectly along the x axis. It just a matter of LMB close to a cross on the grid that contains our objects center then snap to it.

Cursor->Active might not be so helpful in spin modeling. Cursor->Active snaps the cursor to the active element (usually the element that is last to be selected). Try it by selecting several vertices, then do Cursor->Active snap. The vertex where the 3d cursor goes is the active element in your selection.

Spin Tool Parameters The spin accepts two parameters to execute its function. These are Degrees (degr) and Steps.

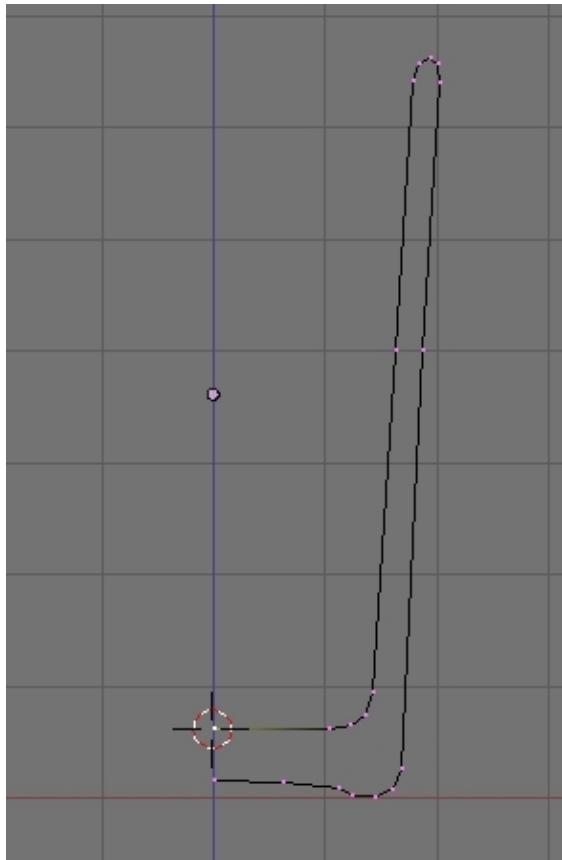


Here our 3d cursor is not at the center of our drinking glass outline. We select one of the central vertex and do Cursor->Selection snapping.



Degrees specifies the angle at which the selected geometry will be rotated from its initial position around the axis. It accepts values in degrees. A 360 degrees value being a whole turn around the axis. Values lesser than 360 produces something like a wedge is sliced from the side of the form.

Steps specifies the number of “copies” of the selected



Result after doing the Cursor->Selection snap.



geometry the spin tool makes that it arranges in a radial array around the 3d cursor. A higher value result in much rounder form.

Cleaning Up the resulting Mesh After using the spin tool, a little up work might be necessary, especially if you specified a 360 degree turn. Though the mesh looks clean to the eye, the tool actually make geometries (vertices) that are in the same blender position as another one. So what might look like one vertex may actually be two. A simple way to clean up our mesh would be to select the whole geometry then activating the “remove doubles” from the specials menu (Wkey(Specials Menu)

-> remove doubles or Wkey->6key). This would remove these problem geometries, finds vertices that are too close to each other and merging them to one.

5.32.3 Post Modeling

After the lathe able parts are done it would be ready to adding non latheable parts. For example a cup may have a bowl-like body, that we could use the spintool. Alas the handle can't be so we had to put it there after. Depending on the geometry added. This geometries could be formed from the mesh itself or be adding in a separate mesh.

5.33 Illustrative Example: Model a Wine Glass

Lets start with a classic spin tool model. Here we go step by step in making a wine glass... or any other latheable models... using the spin tool.

5.33.1 Model half the outline

Using what you have learned earlier, create a simple outline of the wine glass. Just the half part and leave the center part open. As a recap, go first to front view. Select the cube (or if you have no object create one) and press Tab to go to edit mode. Delete all vertices of the object, then using Ctrl+ LMB, add connected vertices. Keep holding Ctrl and clicking until finished. Adjust vertices as required.

5.33.2 Spinning the outline

Remember to align the two end points in the open side. Select the two vertices and scale at the x-axis to zero (Skey->Xkey->0key). Move the 3d cursor to the center by snapping to the selection (Shift+S -> 4key). Then select all vertices by hitting Akey twice.

Now go to top view (Num7). Select the editing button (F9Key) and find the mesh tools tab. Change Degr to 360 for a complete turn and change steps to 20 or higher to form a much rounded shape.

Press the Spin button.

5.33.3 Cleaning up the Mesh

Select all vertices by pressing Akey twice then apply remove doubles operation (WKey->6Key/Remove doubles). Now the model is finished. Make it shade smooth using set smooth (Wkey -> set smooth). If you like, add a Subsurf modifier to the object to make it geometrically

smoother. If dark bands appear, recalculate normals outside (Ctrl+N), to eliminate them.

5.34 Creating Ogg-Theora movies using Blender

Wikimedia Commons requires that movies be uploaded as Ogg-Theora (OGG) files. As of Blender 2.42a, this is not a builtin feature of Blender. To get OGG files from your finished animation isn't difficult, though. However, you'll need additional software.

There are basically two ways to generate OGG files: you can use one of the many fine video editors or you can use special conversion programs. Video editors like LiVES or Cinerella allow you to load your AVI or your rendered frames, manipulate them, and create the OGG file from it. Please refer to the editor's documentation on how to achieve this.

A disadvantage of a video editor is they are huge pieces of software, duplicating functionality that you already used when you created your animation file/s with blender. It's actually not necessary to install a video editor just for converting your animation to OGG Theora format.

5.34.1 Converting saved frame picture files to Ogg Theora

It's actually possible to convert frame pictures that you saved before to OGG format movies. The [ffmpeg2theora](#) software package, which is available in source or binary for all relevant systems, is capable of batch-processing files into an Ogg format movie. For example, if your frames were saved as PNG (with filenames filename001.png, filename002.png, etc.), you could convert them to a soundless OGG file with:

```
ffmpeg2theora filename%03d.png -o output.ogv
```

Sound is possible too, as well as being able to set the quality and framerates. Consult the [ffmpeg2theora](#) documentation for more.

5.34.2 Converting AVIs to Ogg Theora

[ffmpeg2theora](#) can also convert AVI movies to OGG. Usage example:

```
ffmpeg2theora --optimize my.avi
```

5.35 Creating animated GIFs using Blender and Gimp

This tutorial will guide you through how to make a simple animated Gif Using Blender and Gimp. This is useful

for creating Avatars for forums etc. This tutorial assumes basic knowledge of blender and Gimp, see the [basic animation tutorial](#).

To start off you will need an animation, this usually should be no longer than 25 frames long.

1. Open Blender. and delete the default cube and add a UV sphere, the default settings for the sphere work fine.
2. Now set the camera size in the scene buttons (F10) to 50 by 50 (I am creating an avatar for deviant art where the required size is 50 by 50 pixels, you may change this if you want)
3. set the frames to start at frame 1 and end at frame 20.
4. select the sphere and insert a LOC keyframe at frame 1 and 21, then go to frame 11, move the sphere and insert another LOC keyframe. this will create a looping animation.
5. Set the image type to PNG or JPEG (it doesn't matter) and render the animation.

Now to combine the images into an animated GIF using Gimp.

1. Open the first image with gimp.
2. Now click File-> open as Layer or press "Ctrl-Alt-O". Select the next frame and it will be added as a new layer. Repeat this for all of the images, or select all of the images by pressing "Ctrl-A".
3. If you press Filters->Animation->Playback it should play the animation. It will probably have a low frame rate making it "choppy". this will be fixed in the next step.
4. Change the frame rate to 40 ms(25 frames per second). Choose file->Save as-> "Name".gif, then choose Save as Animation.

Note: Don't know wheter this is an additional feature of Gimp 2.6.8 or not, but i found out following: In Gimp 2.6.8, within the export wizard, you have to choose option: -save as animation- first and next at the option: -single picture where not mentioned- the option: one single picture per layer. -Sorry i use the german version of gimp-



You should end up with something like this:

Alternatively, you may try using addons to create animated Gifs directly from Blender. 2 options include [Spritify](#) and [Bligify](#)

5.36 3D Tiling Backgrounds For The Web

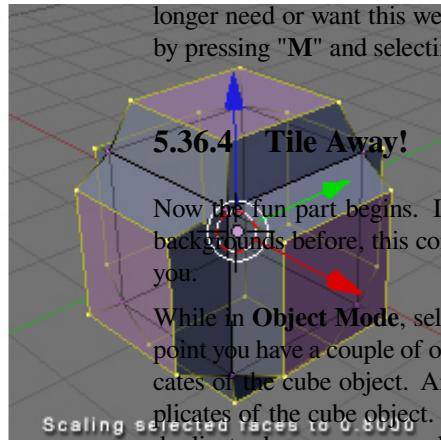
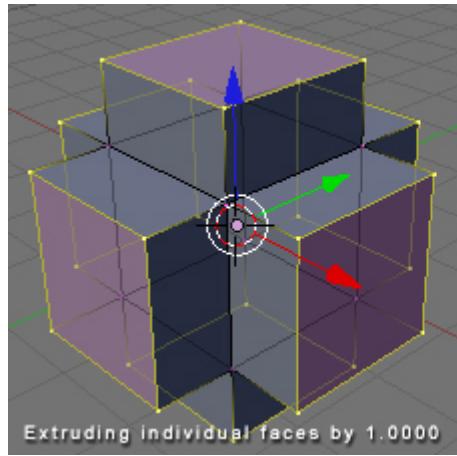
5.36.1 Overview

This tutorial will guide you through the process of making 3D tiling backgrounds for use with web pages, your desktop, or anything else for that matter. We will be using Blender and a graphic editing program such as Adobe Photoshop or GIMP. I will provide detailed explanation to cater to beginners, however more experienced Blenderists can probably get away with just following the picture diagrams provided. You may also find the diagrams useful if English is not your native language, or if you hate reading instructions.

5.36.2 Create The Object You Wish To Tile

Start up Blender and look at your lonely cube. We will use this cube as a starting point for the sake of demonstration, but feel free to use any shape you like. Press "Tab" to enter **Edit Mode**. Press the "E" key on your keyboard and the **Extrude** menu will appear. Select "**Individual Faces**". As you then drag your mouse cursor you will see some numbers move in the bottom left in the window and text saying "Shrink/Fatten". I recommend setting it to 1.0000 to keep it simple. Hold down the "Ctrl" key while dragging to do so in set increments.

Next, with the six faces of the cube still selected, press the "S" key to scale those faces. I recommend scaling to 0.8000.

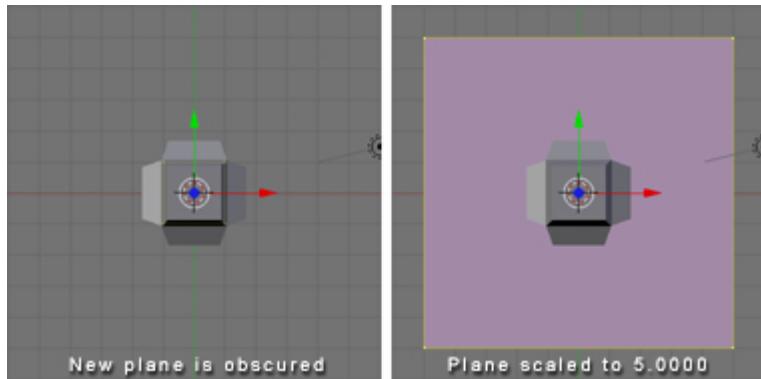


Good. Now that we have created our object we will prepare to tile it.

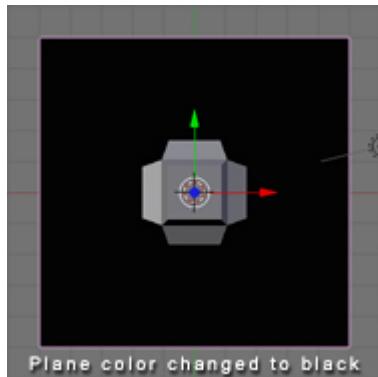
5.36.3 Specify Your Tiling Area

"Tab" back into **Object Mode** and press "NumPad 7" to go to the Top View, or manually click "View" at the bottom of the window and select "Top".

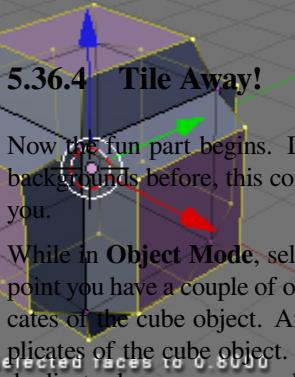
Press "spacebar" and from the menu that appears select "**Add** > **Mesh** > **Plane**". The plane has appeared, but is obstructed by our cube object, so press "S" to scale the plane to 5.0000.



You'll notice that I've colored my plane black. This is only to make it easier for you to see. You need not bother with this as we will be deleting the face of this plane shortly.



We now have an easily visible boundary representing the area which will be tiled. If at any point we decide we no longer need or want this we can move it to another layer by pressing "M" and selecting a layer to move it to.

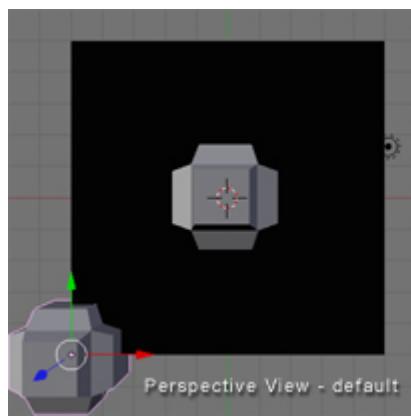


Now the fun part begins. If you have created 2D tiling backgrounds before, this concept will be very familiar to you.

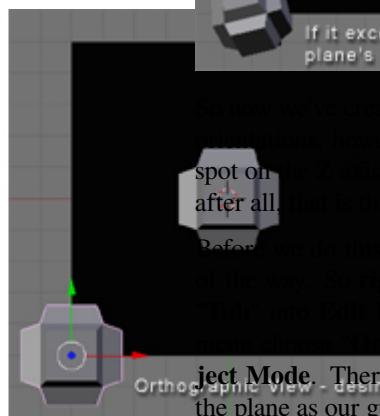
While in **Object Mode**, select your cube object. At this point you have a couple of options. One is to make duplicates of the cube object. Another is to make **linked** duplicates of the cube object. I recommend making linked duplicates because you can later edit the mesh of one object in order to change the mesh of all the duplicated objects. This can be especially useful if you later plan to animate your tiling background (*careful not to distract from the foreground*). It is also useful if you want to experiment with new designs easily without having to re-place all your objects.

With your cube object still selected, press "Alt+D" to make a linked duplicate. (*Non-linked duplication is done with Shift+D*) Now hold down "Ctrl" to move in set in-

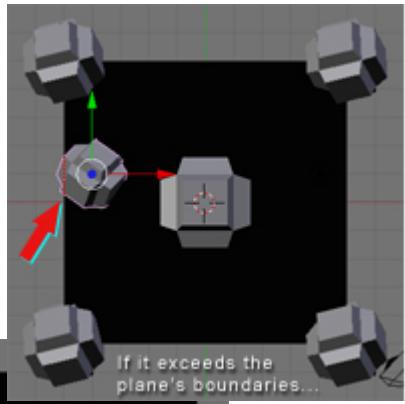
crements and move the object to one of the corners of our plane. Make sure the center of the object (represented by a pink dot) is aligned with the very corner of the plane. At this point you may notice it's somewhat difficult to tell whether the center of the object is exactly on the corner of the plane, and that's why we are going to press "**NumPad 5**" or click "**View**" and select "**Orthographic**". For all practical purposes we can spend the rest of the time we are building the tiling pattern in Orthographic View because this view is essential for tiling.



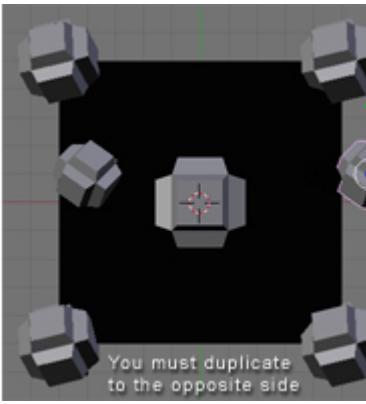
Perspective View - default



Orthographic View - default



If it exceeds the plane's boundaries...



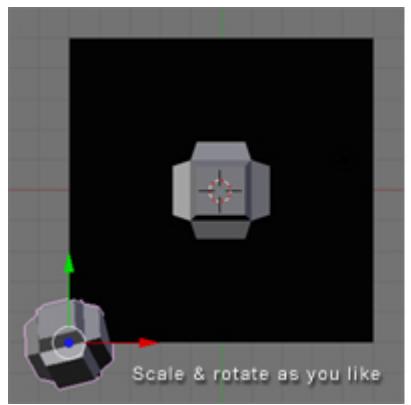
You must duplicate to the opposite side

Some objects I created a few objects of different sizes and shapes. However they have all been along the same plane. So let's give our pattern some depth - after all, that's the joy of working in 3D!

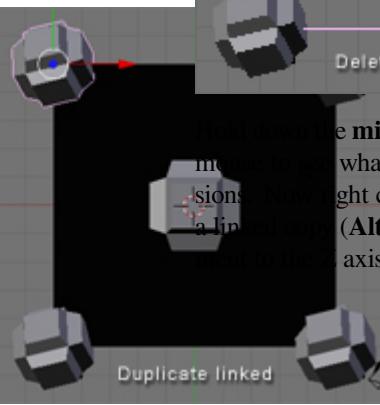
Before we do that, we'll want to get our guide plane out of the way. Right click on the plane to select it, then switch to **Edit Mode**. Press "X" and from the **Erase** dropdown choose **Only Faces**. Now **Tab** back into **Object Mode**. There! We now have only the segments of the plane as our guide.

Now with your duplicate object still selected, scale it ("S" key) so that is somewhat smaller than the first. Then rotate it using the "R" key. While rotating you can constrain to a particular axis by typing "X", "Y", or "Z" respectively.

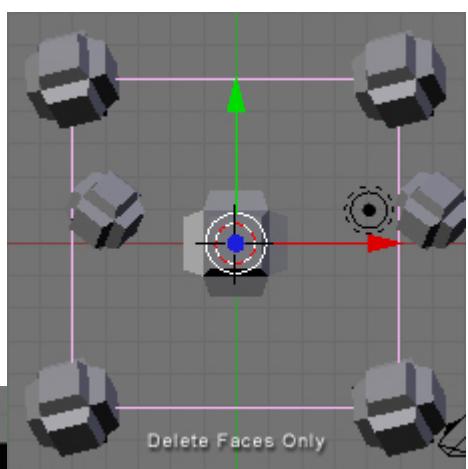
We will now make linked duplicates (**Alt+D**) of our rotated and scaled cube object and place one in each corner exactly on the grid (remember to hold "**Ctrl**" while dragging). If you place an object in one corner, you must place it in all four corners because corners touch both the X axis and the Y axis.



Scale & rotate as you like



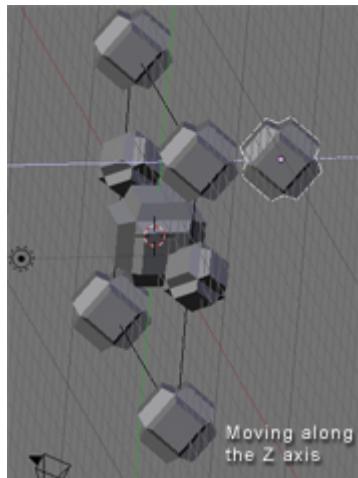
Duplicate linked



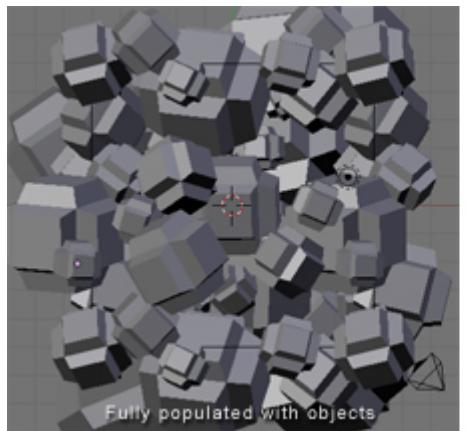
Delete Faces Only

One last tip: use the **middle mouse button** while dragging the mouse to see what our design looks like in three dimensions. To do this, right click one of the cube objects, duplicate it (**Alt+D**), and press "Z" to constrain movement to the Z axis.

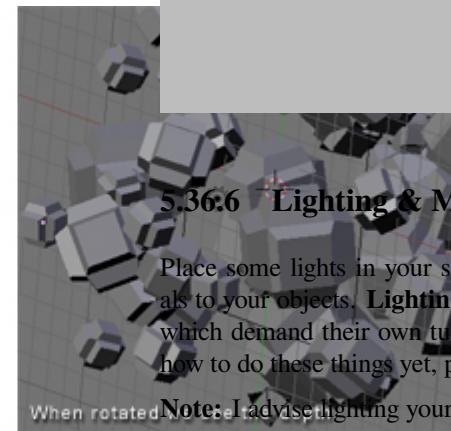
Now as you will notice in the following diagram, I made an object that is crossing over the edge of our plane. This is good. This is how to make natural tiling backgrounds. However! Whenever we do this we must be absolutely certain to do the same on the opposite side, or else our backgrounds will not tile properly.



Navigate in this way along the **X** and **Y** axis as well and once you've found a good spot, **scale** and **rotate**. Remember to make a duplicate on the opposite side whenever you cross the outline of the plane guide. You can go on like this and populate your tiling pattern with as many objects as you wish. The grid is your friend during this process, so you should always have your finger on "**Ctrl**" while moving the duplicate of an object that crosses a border of the guide plane. Also, any object that will "tile" across the border must be at the same point on the **Z** axis as its counterpart on the other side of the seam. Press "**NumPad 7**" periodically to see where you are from a 2D standpoint. This is the view from which we will eventually render the image, so this perspective is the one that counts. Beginners, remember that **mouse wheel** zooms in and out, and "**Shift+middle mouse button**" allows you to "drag" your way around.

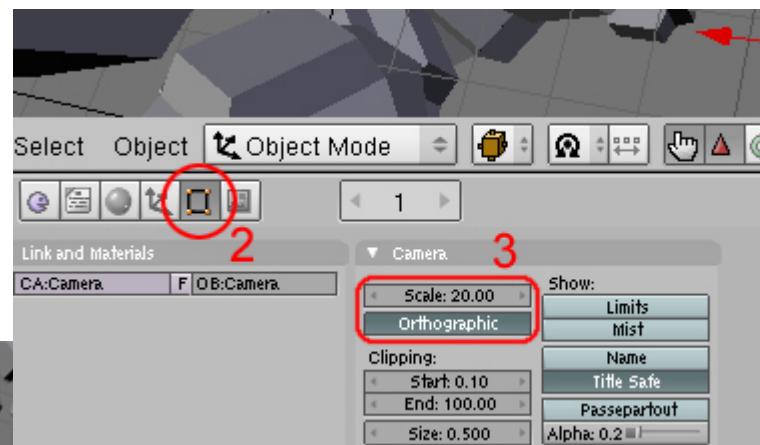


Keep in mind that in **Orthographic View** we do not perceive the depth of objects. Those far away and those near by appear to be at the same distance. This is what allows us to make a tiled image, but it also limits the apparent depth of the scene. We can compensate for this by scaling down the objects we want to appear further away. Or, as in this example, we can just make each object a different size and place some in front of others.



5.36.5 Camera Settings

We will now position the camera directly over our cluster of objects. Press "**NumPad 7**", center the view on the cluster using "**Shift+middle mouse button**", and zoom out a couple notches with the **mouse wheel**. From the **3D View Window**'s menu, click "**View**" > "**Align View**" > "**Align Active Camera to View**". Now you may be wondering why everything appears to be distorted. This is because when we changed to the camera's perspective, the view automatically reverted to **Perspective View** because by default the camera is set to that view. But just as we changed to **Orthographic View** in the 3D View Window, we can change the view of the camera as well. While in **Object Mode**, swivel the view until you see the camera. It is represented by a pyramidal wireframe with a black triangle atop the opening. **Right click** to select it. Now press "**F9**" or click the **Editing** button (it's icon is four vertices joined in a square). Now in the **Camera** panel you will see a button labelled "**Orthographic**". Press it. Above the button is a value labelled **Scale**. Set it to around 20. (See figure below)



5.36.6 Lighting & Materials

Place some lights in your scene, and add some materials to your objects. **Lighting** and **materials** are subjects which demand their own tutorials, so if you don't know how to do these things yet, please consult the Wiki.

Note: I advise lighting your scene pretty evenly in order to make the tiling seem contiguous.

Note: You could also only use Sun lamp, so the light is clean all over the mesh!

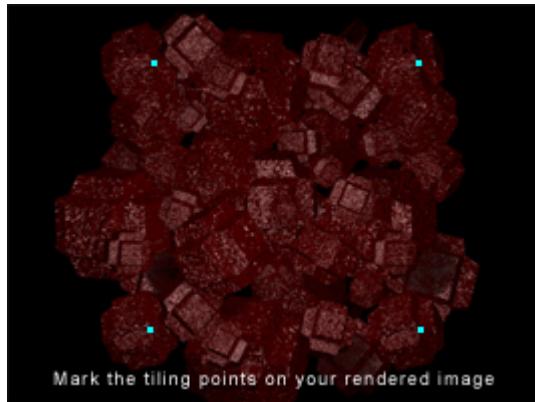
5.36.7 Rendering

Now that you've presumably got your lighting and materials as you want them, it's time to render the scene. Under **User Preferences** click "**Render**" > "**Render Settings**" or just press "**F10**". In the **Format** tab you can choose the dimensions of your rendered image, the file format

you prefer, and the quality. I'm going with 800x600 at 100% quality.

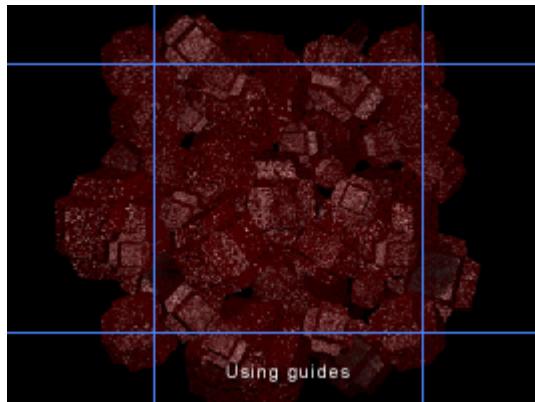
5.36.8 'Shopping

And it's time to begin post-production. Open your rendered image in your preferred image editor. I will be using Photoshop in this example. Now find and mark the four corners of your tile (on a new layer of course). Just eyeball it.



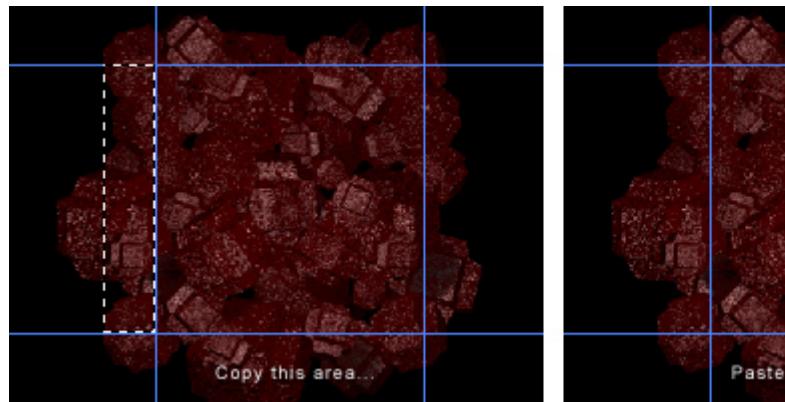
We will use guides to mark the tile boundaries. In Photoshop guides can be inserted by selecting "View" > "New Guide...". Guides can be positioned with the **Move Tool**. You can zoom in and make sure your guides are accurately positioned by holding "Ctrl" while typing "+" or "-". You can now hide or delete the layer with the markers, as that was only to help us get accurate guides.

Should you decide to make an animated tiling background I recommend layering all your frames on top of one another in your image editor before editing in order to keep the position uniform. You can then animate them using Adobe ImageReady (comes with Photoshop) or find decent low-budget and occasionally free GIF animation programs online.

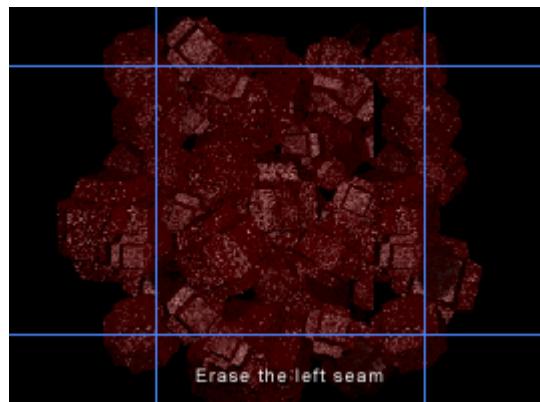


Now select an area just *outside* of the guides with the **Marquee Tool**. Make sure "Feather" is set to 0px and that the marquee is rectangular and **Style** is set to **Normal** (meaning no Fixed Aspect Ratio or Fixed Size). Also, it will help you to have **Snap** enabled, so go to "View" and make sure it is enabled for guides.

Once you have selected the area, copy (**Ctrl+C**) and paste (**Ctrl+V**). Then drag to the right using the **Move Tool** and that piece should snap to the inside of the rightmost vertical guide.



Select the **Eraser Tool** and choose a brush size. I'm going with a diameter of 65. You'll want to use a soft gradient-like brush for this. Now erase the left edge of the newly pasted selection so that it blends into the picture. Don't erase any of the right edge.



Press "Ctrl+E" or alternatively, select "Layer" > "Merge Down".

Alright, so that procedure you just performed copying from left to right - now do it bottom to top. Once again, erase the edge. Do a **Save As**.

Finally, select the square center area that will become your tile and do "Image" > "Crop".

Now scale your new tiling background to a web-friendly size. **Save As** a JPEG, GIF, or PNG. **You're done!**

Note: If your pattern still isn't tiling quite perfectly, load the version with the guides again and repeat the copy-paste-erase process. Remember to merge down your layers each time or the corners may not tile correctly. If still no joy, try tiling the background in a web browser, taking a screenshot, pasting that into your graphics editor, and touching up from there.

Here is the final result:



car.blend, or car.JPG

Next time do “Save As” then press the + (plus) key which will advance the blend file by 1 every time it’s pressed. example: car.blend becomes car.1.blend. press again car.2.blend and so on.

The - (minus) key will subtract one. I’ve gotten into the habit of saving frequently. Yeah I know there is now the undo feature but I like this better because it gives you a history in case you need to back a few levels of a build. You get a saved version at the level you choose.

Note: Blender automatically defines the number... meaning it does not need to be in any position. For instance: If you have a file named 001starport.png or .blend or whatever, pressing the + (plus) key will automatically name it 002starport.png. If you want to name it starport1.png, it will change it to starport2.png.

Two rules: The filename has to have a number. It can be 0, or 1, or 3.141569.

If the file number is a negative, pressing + (plus) key will increase the “magnitude” of the negative number. I may have used magnitude wrong, if so, I mean pressing the + (plus) key will make -0.04 drop to -0.05. The - (minus) key will bring you only to 0, and then it will start to eat itself up.

5.37 Cool Things That Aren't That Obvious in Blender

Cool things in blender that aren't that obvious. Useful tips and tricks in Blender.

5.37.1 Attribution

Many of The Following Tips and Tricks have been contributed by members of the [CGsociety.org](#), a Public Society for Digital Artists. The tips have been extracted from a [CGsociety thread](#). They have been roughly edited to improve readability.

5.37.2 File Browser Functions

Delete, Move, Rename, and Make Directory

When you are inside a file browser for loading or saving something and you want to create a new directory, just add the name to the path on top of the window and confirm 'Makedir'.

you can also delete(x), move(m), or rename(n) a file. you can do action on multiple files by selecting with right click

Preview images when loading them as a texture

Whenever you are loading images as a texture, you can hold down **Ctrl** while clicking on the 'load'

Trick for Creating Quick File Revisions

After you have saved a blend file or an image you can then save it in progression that is:

Open Recent

Control O

5.37.3 Object/Vertex Manipulation

Constrain Movement to one Axis or to a plane

when moving objects/vertices or set of objects/vertices ([G] key) if you move in the direction of the global X axis (Up/Down) and then press the **MMB**, movement will be constrained to only move in the X axis, or if you move the vertices in the direction of the Y axis and then press **MMB** it will be constrained to move only in the Y axis. The same is true of the Z axis.

You can achieve the same effect by using the X, Y or Z keys while in grab mode. You simply have to press X key, Y or Z, once to lock to X Y or Z global axis,*

To Constrain the Movement to two axes (a plane): Press **Ctrl+X** to move in the Z-Y-Plane. **Ctrl+Z=XY**, **Ctrl+Y=XZ**.*

Alternatively, Select scaling mode and select the axis not to scale with the selecting button at the same time as you hit **Shift**.

- in every case You can hit the X,Y or Z button again to constrain movement to a different set of axes. Normally this different set of axes is local. However

you can change the identity of this set to global, local, normal, or view by pressing **Alt+Space**. which cycles through the different identities.

Note: All of these shortcuts work with scaling and Rotating as well

Shrink/fatten mesh in direction of vertex normals

When you're mesh-editing, **Alt+S** will shrink/fatten the mesh selection in the direction of the vertex normals.

Vertex Parenting

You can parent object to a Mesh, in that case you are parenting to the center of the mesh.. BUT if the mesh is translated somehow (let's say by an armature'**[S]** pose) the center remains in the same spot, and thus the child object doesn't receive any transformation at all.

To solve this, you can parent the child object to a vertex (or a face) within the mesh, and any transformation that the vertex receives is passed to the child.

There are only 2 options, to parent to any 3 vertex within the mesh or to parent to just one vertex. If you parent to 1 vertex then only location information is passed, with 3 vertex all transformations (rotation, location and size) are passed to the child.

How to do it? Starting in object mode select the child(ren), hold **Shift** and select the parent, enter edit mode, select one or three vertex, press **Ctrl+P**. That's it!

Work around to welding Verts

For Edge loop (Verts) position both loops together as close as possible then hit W then 4 (not on numkey pad). You can adjust how far the effect of collapse can go in the Edit window (a button on the right labeled Limit: ***, where * is a number).

As for individual Verts, e.g. Two vertices welded to become one, select both Verts, scale until they are very close then hit W then 4.

you can also do this with the snap combo

select the vertex you want to weld together **Ctrl+S**, Cur -> Sel **Ctrl+S**, Sel -> Cur [**W**], Remove Doubles

Also, whenever Blender pop up a menu with different options, you can just type in a number to choose one of the options (use the numbers not on the **NumPad**)

Make individual objects the camera and Change them back

If you select certain objects and press **Ctrl+0(zero)** it will make them the camera. I use it all the time to align spot-lights.

Select your camera and hit **Ctrl NumPad 0** to make it the active camera again

Ordering Meshes in Vertex Groups

If you are preparing to skin your meshes and you are ready to create the vertex groups, you should pay attention to the order in which you create them, because once they are created there is no way to re-arrange them on the vertex-group list.

That means that if you are a ultra-by-the-book person and you would like the vertex group alphabetically ordered on the list them you must create them in alphabetical order.

.. Or, if you would like them to be ordered according to their function (shoulders, then arms, then forearms, then palms, etc. you must create them in that order.

This may all sound like a stupid thing to care about, but If you have a character with 39 vertex groups you may quickly find that when one of them needs fixing it is a little difficult since they were randomly created.

Position and scale along face normal,

-**Shift V**: Position camera along face normal, **Alt S**: scale selected vertices along face normal

Using Fake Users

pressing **Shift+F4** will turn the window into a "Data Select Window" where you can assign and unassign fake users to almost everything by selecting the name and pressing the F key

Creating a fake user allows you to keep useful data blocks (materials, textures, base meshes) at hand even if they are not linked to an object. You can use it to set a default material that would have the shader you like best.

Align a selection of vertices on a plane

If you want to perfectly align a selection of vertices on a plane, you just have to follow these little steps:

1.) Before you are selecting the vertices you want to align, position your 3D cursor in the plane that you want to align to (you could select the 4 vertices of a big plane and hit **Shift+S** / Cursor->Selection for example, but you can position it anywhere)

2.) Now select those vertices you want to align

- 3.) Choose “3D Cursor” under “Rotation/Scaling Pivot”
- 4.) Now with the [S]-key start scaling mode, hit the key of the axis you want to move the vertices on (X,Y,Z)
- 5.) Holding down the **Ctrl-Key**, you can now move the vertices in one line towards the cursor, until the value for the chosen axis is 0.000. Alternatively, just enter, using the keyboard, the scale factor you want (0 in this case) .
- 6.) Hit LMB. The vertices are perfectly aligned along a plane through the 3D cursor.

This even works very well while in perspective view mode, so you can align on the fly and don't have to switch to front/side/top view all the time.

Welding Vertices

You can weld vertices by selecting them in edit mode and pressing **Alt+M**.

View wireframe of hidden Verts

to view the wireframe of hidden Verts, make sure you are in WIREFRAME MODE and then turn SubSurf on and change the level to 0 If you already knew about this then

Select a true loop

shortcut is **Shift + Alt + right Button** of the mouse, serves to select true loop, in vertices as in edges like in faces.

Selecting one object from a single mesh comprised of multiple objects

If you have more than in Edit Mode, you can place your mouse cursor next to one of the Verts in the desired object, then press the “L” key to select all of the Verts linked to that one. “Alt+L” reselects in the same manner.

Precise Zoom and Select/Deselect

Selecting: If you Hold down the **Ctrl+LMB** (left mouse button) and drag the mouse, it will allow you draw a selection as opposed to using the B button which gives you a square.

Deselecting: To draw an area to deselect, Hold down the **Ctrl+ Shift+ LMB**(left mouse button) and drag the mouse,

Zooming: Hold down the **Ctrl+MMB**. Move your mouse vertically to can get a more controlled zoom versus scrolling the Mouse wheel.

This feature may not be present in 2.37 or earlier versions.

Mouse Gestures

Left click and draw:

- a straight line - moves the selected object.
- a circle - rotates the selected object (note: this must be drawn fairly circle-like).
- a V - scales the selected object.

Selecting Obscured or Hidden Objects

Say you are in front or side view and you want to select an object, but it is obscured or hidden behind other objects. If you press **Alt RMB** over a group of objects, a menu will be displayed in the 3D window allowing you to pick the object you wish to select.

Select or Deselect Multiple Vertices

In Edit mode, when you click the **RMB** near a vertex that vertex (face or edge)will be selected, **RMB** again will reselect. By holding the **Shift** key this will allow you to add each selected vertex (face or edge) in that highlighted group.

Pressing the U key (Undo in Edit mode) will also remove the last selection you made.

Alternatively, you can Press B and then draw a box with **MMB**. Anything caught in the box will be deselected. Also works with BB and the draw selection. Draw with **MMB** and you reselect.

Selecting multiple items

You can hold **Shift** and use right mouse button to select multiple items to append.

Measuring, length, distance on an object.

Hit “F9” (editing), you should have split (2) windows. One “3d” the other “buttons” go to the Mesh Tools 1 panel and press the Edge Length, Edge Angles and the dimensions will appear on your selections in the 3d view. This description makes no sense and should be changed.

Adding Connected Vertices

In Edit Mode, if only one vertex is selected, pressing “E” will add a vertex, on a freely defined place, connected to the selected one. As will holding **CTRL** and left clicking the mouse where you want the new vertex to be positioned.

Note: It must be an Image texture and in wireframe mode to be visible.

(2.37a)

Recalculate Normals

Ctrl + N = Recalculate normals outside (you might have to select faces before doing so) **Shift + Ctrl + N** = Recalculate normals inside

These two hotkeys are useful when you extrude some edges and see a kind of seam in between (due to normals pointing in different directions).

Then, after selecting an edge, **Ctrl + NumPad(+)** selects the face associated with this edge. **Ctrl + NumPad(-)** deselects the face.

Create a Quad from two Tris

Alt + J when having two Tris selected makes a Quad.

Remove Doubles

To Remove Doubles use hotKey: **W**. You can adjust the “limit” option so that “Remove Doubles” has more or less tolerance. This option is located in editing window under the mesh tools panel, (i.e. weld vertices that are further?)

Combine edit levels on a mesh.

When in Edit mode for a mesh (TAB key) you can choose the level that you wish to edit at. At the bottom of the 3D window, there are four buttons. Vertex, edge, face & back-face cull.

By default the vertex level is selected, if you hold **Shift** and press the edge button, you can use both at once.

Change Select Mode

to change select mode (vertex, edge or face) you can press **Ctrl+Tab**. But this way you can't use the **Ctrl+Key** to ADD the select modes.. Still could be useful, if you don't have a header for the window you're working in..

Precision Warping

When using the warp tool (**Shift+W**) you may find that there are times when you have trouble perfectly closing a 360-degree loop. This is because Blender will warp based on the total width of the selection, which may not necessarily be what you want.

Getting around this is simple, just select two verts to denote the new endpoints, duplicate them (**Shift+D**), scale the two verts times two, so they are just twice as far apart. Then select what you want to warp plus those two marker verts, and warp 720 degrees.

When done, click a vert on your mesh and type **L** to select everything linked to it. Then type **H** to hide it. Once you've hidden all of your mesh, all that remains are the two marker verts. Do a Select All (**A**) and type **X** to delete them. Now unhide your mesh using **Alt+H** and you have just the warped mesh, extra guide verts are all cleaned up.

Precision Cutting

The Knife Tool can actually be quite precise if you take advantage of the snap feature. Press **K**, choose “**Knife(Exact)**” and then hold the **Ctrl** key while choosing where you wish to cut and you will find that the path to be cut will now snap to nearby vertices.

Keep in mind that the vertices being snapped to don't have to be the ones you are cutting. Say you want to cut the midsection out of a UV sphere and you want the cut to be two grid units in height. While viewing the side of the sphere, add a plane, and scale it to be two grid units tall, and wide enough so that it extends beyond the sphere. Now select the vertices of the sphere (because what we have selected is what gets cut) and when you cut it, snap to the four verts of the plane. Now hit **Enter** to apply your perfect cut.

Using guide geometry such as a plane to cut other geometry “cookie-cutter” style can be extremely useful when accuracy is needed. Remember to align your view before cutting, since your view will determine the angle from which the geometry is cut.

5.37.4 Working with Meshes

Turning Sub-Surfed Mesh into Normal Mesh

If you have a sub-surfed mesh you can turn THAT sub-surf mesh into a normal mesh. Just select the sub-surfed mesh and press **Alt+C** (Conversion).

How to remove (numb) black spots on a Mesh

If a Sub-Surfed mesh becomes (numb) black on some places, that's because of the normals. Select all and press **Ctrl+n** and then confirm. now it should look prettier!

if the above solution does not work , save your Blend file, Quit Blender then restart. Use **Ctrl O** to open the last file and your mesh will have returned to correct shaded state.

Select all holes in a mesh and fill them

Shift+M* selects all Non-manifold edges/vertices (holes) in a mesh **. Then all you have to do is hit **Ctrl+F** to auto-fill those holes with “beauty fill”.



Shift+M is an alternative shortcut for 'Select Non-manifold'. You'll find this in the select menu when in mesh edit mode. The listed short cut there is **Ctrl+Alt+Shift+M**. **Shift M** is obviously a lot more comfortable on the fingers though!

• •

Although 'Non-manifold' usually refers to holes in your mesh, it is not necessarily only holes e.g., an edge with three faces coming out of it is also a non-manifold edge!

Fill in four or fewer vertices

Select the vertices and press **[F]**, this will fill in the empty space around them.

To clean up a filled in space, select all the vertices for the area, and press **[F]**. Choose OK to make FCon. For example: add a plane in wireframe mode, extrude it several times, select all vertexes, then hit **[F]**.

5.37.5 Animation

Animation Preview in all windows at the same time

It is well known that **Alt+ a** is for previewing an animation on the 3D window. But that's not all of it. Divide your screen into multiple 3D Windows, each from a different point of view.

Press **Alt + Shift + a**

Enjoy!!!

If you have an Action/Ipo Window and 3D windows open, and you issue the **Alt + Shift + a** command from the Action (or the Ipo) window, it will animate both (the action and the 3D) in sync!! Great for visualization of Ipo's effect on your model.

Choose animation mode, Convert mouse movements to IPO

-T in IPO window: Choose animation mode, i.e., linear, bezier, constant -**[R]** in IPO window: Record mouse movements, and convert to IPO

Animate procedural textures

To animate procedural textures, press 'i' with the mouse pointer in the materials window, and select the type of Key-Frame you wish to set in the pop up menu. advance a few frames, tweak your materials, and set another key frame.

5.37.6 UV Mapping, Particles and Texturing

Blender color picker

Blender has a PhotoShop-esque color picker. Simply click on the color preview next to the sliders to use it. Hit enter when you have the color you want.

Saving your face groups selections

Regarding UV mapping and Face Groups Selections there seems to be the general misconception that you can't save your face groups selections on Blender.

Most people already know that from within the Face Select Mode (Potato Mode) you can switch into Edit Mode and whatever selection you do while in Edit Mode is passed back to Face Select Mode when you exit the Edit Mode.

Well, Did you ever wonder why Material Index Groups (that are nothing more than face groups with a common material on them) have those little 'Select' and 'Deselect' buttons there? Sure they come handy for later modification of the material index but that is not all about them.

Do this: Before starting the UV unwrapping job, cut your mesh by creating as many material indexes as you need, you can even assign each one a different color so you can be sure that there is no face orphan. Once you have the mesh all cut and sliced (so to speak) you enter in Face Select Mode, then switch into Edit Mode, select the index containing the faces you want to unwrap, press 'Select', leave Edit Mode and Voile! You now have an entirely useful face group waiting for you to unwrap. No more manual (and imprecise) face selection is needed.

If you later need to change the mapping of those faces, don't fear. Just make sure there isn't any face selected on Potato Mode, do as you did first (enter edit mode, select the index, exit edit mode) and there are your very same faces selected again with the UV mapping you already assigned to them.

Note: Another benefit of have precise face selection groups is that, initially, you don't have to worry about UV coordinates overlapping, since you know have the way to select ONLY the faces you want to. For example, you unwrap all your faces by groups and when you are done you can start thinking about scale and position within the texture map, not like before when you have to solve those things as you go.

Bulk Texture Change

Consider a scenario in which you have a scene with a 100 mesh objects, and 50 of them have one texture and 50 of them have another.

If you want to change the texture of the first 50, but don't want to change each individually, do the following. Add a Plane out of the view of the camera. Add your new texture (Material) to the plane. Then use the "Copy to material Buffer" button in the header of the Material buttons. Select one mesh object of the same sort that you want to change, open Material buttons and Paste from Material Buffer.

All the mesh objects with the same texture will now have the new texture.

Alternatively, If you have a material that you want to apply to a lot of objects at once:

1. Select all the objects you want to apply the material to.
2. Apply a material. (this only applies to the last selected object)
3. press **Ctrl+L** > Materials. (this links the material of the last selected object to all the other objects)

Negative Meta-Balls

Add>Metaball as usual. Exit EDIT mode and Add>Metaball. This time before you exit EDIT mode, hit the Negative button in the EDIT buttons window. Then leave edit mode.

If you move the negative Metaball around, you can see the effect it has on the positive metaball.

Be careful though, as negative metaballs are not displayed in the same manner as positive metaballs, you will only see the Pivot point not a meta-mesh.

This is a little test you can try to see the amazing effects negative metaballs can produce.

Make 1 Metaball, make it big. Place three spheres (UV) inside, make them emit particles, one 100, one 200, one 300 particles. Parent three negative metaballs, one to each sphere, and use dupli-vert on each sphere. Make the 100 duplicate metaball quite big, the 200 medium and the 300 small. Hit **Alt A** to run animation in 3D window.

(Click here for this author's negative-metaball thread... oops, it's not there anymore, well, not the AVI anyway!)

(Click here for this author's negative-metaball AVI.)

Maintain the UV layout when moving/scaling/rotating UV co-ords.

When you have the UV image/editor window open and have loaded an image you want to UV map to a mesh, click on the UV tab in the header bar and turn off 'Snap UV to pixels'.

This will help to maintain the UV layout when moving/scaling/rotating UV co-ords.

5.37.7 Rendering

Tricks, related to the view ports and the render buffers

First. Switching among screens

So you have your screen made off the 3D window, the buttons window and the info window... but you are doing some fine tuning to the mesh in two places simultaneously, and they both need to zoom in the 3D window. You could scroll or zoom out, translate the view and zoom in again. None of them an elegant solution.

Another situation. You are working on a model and are using an image for reference. You are not tracing over the photo, just take a look at it often to make sure you don't deviate to much from the concept. So you open the photo in a 2D program and keep switching back and forth from Blender.. or you have the photo open in an image window and keep maximizing and minimizing the window... another hassle

Worry no more!!! Blender can handle multiple virtual screen (ala Linux) and you can come and go from them with just one key stroke.

Just press **Ctrl+Left Arrow** or **Ctrl+Right Arrow** (for all of you OS X users, add a **Shift**) and you are switching screens. Go ahead! By default EVERY .blend file comes with 3 screens ... and of course you can add/delete as many as you see fit.

the magic button to add or delete screens is right beside the Tools menu, up there in the info window.

Using the render buffers

Ok, so you set your scene and press RENDER, a nice window comes up and you see your hard work coming to existence (that's the default behavior, if you change it on the display buttons then this may not work for you).

Do you realize that the window containing your render image is also a render buffer? Actually they are 2 buffers for you to play with. Whenever the render window is open (and you can re-open it by pressing F11 without having to wait again for the render) if you press the J key you can switch from Buffer A and B. (the last active one is what you save when you press F3). You can even switch buffers in the middle of a rendering (but I advice against that when rendering very complex scenes, you have been warned!)

The cool thing about having two separate render buffers is that you can have instant before-and-after images for things that you change in the scene. For example you are searching the perfect position for a light source in a scene, you place it and do a render, place the light in another position, switch to the second buffer and do a new render. Now, with the render window open, just press J to see how the change on the light's position influence your scene and that makes your decision easier.

By the way, the render window can be zoomed (by the normal ways or by pressing Z) to do a closer inspection of the image.

Render window Tricks

To zoom into the render window, use the **ZKEY**.

To find out what the (Red, Green, Blue, Alpha) values of rendered image are, left click and hold the button of your mouse.this will reveal the RGBA values of the pixel below the mouse cursor. i.e. R 127, G 255, B 13, A40. The values will appear in the bottom left corner of the render window. You can also hold the button and drag the mouse around. This will display the values of the pixels your mouse pointer passes over.

With the render window open, you can press **AKEY** to view an alpha version of the image. Press **AKEY** again to go back to the normal colour view.

To do a before/after comparison after making a change, hit **JKEY** to switch render buffers, then hit **F12** to render. In the render window, use **JKEY** to alternate between the previous render and the current one, so you can easily see the differences.

Working while you Render

If you use a X/X11 based window manager, you do not need to watch blender while it renders, go to a different virtual desktop. Blender doesn't have to keep X informed of what's going on and rendering speed may increase.

Border Rendering

In the rendering buttons find the buttons marked “Border” and “Crop”. If you depress “Border”, you can get a rendering of any part of what the camera sees. Just do the following :

Go into a camera view using **NumPad 0**, press **Shift+B**. Then, mark the limits of the rendering as you want them using LMB. Next, render the usual way and the section you marked will be rendered first, then it will be integrated to the complete rendering. If you wish that *only* the chosen section would be rendered then click on the button marked “Crop” also.

Creating a cluster of particles which takes little time to render

Render some particles, and make the picture into an alpha mapped tga in GIMP or whatever graphics editing application you use.

Load the image onto a plane. Adjust the alpha settings accordingly. Parent the plane to the emitter. Press duplverts.

You now have a cluster of particles which takes relatively little time to render. Of course it doesn't have to be an alpha map of particles. That's entirely up to you.

Alpha from render view

When cut & pasting stuff from render window to {insert your favorite image editor here} using **Alt+ PrtScr**, cut and paste the render first, come back to render window and press “A”. It changes the view to alpha and you get black & white mask to cut the background nicely in the {again, favorite image editor}. Nice when you do testing in low res.

View alpha texture as wire.

Ctrl+d in 3d very usefully for preview without rendering.

Also, if you have an object (works best on a mesh plane) with an image texture, you can use **Alt+V** key outside edit mode. This will adjust the object's size values so that the image won't be stretched when projected.

5.37.8 Miscellaneous

Restoring your “lost” work.

If you go to the temp folder where you have Blender save its temp .Blend files, reload the most recent one and this will save you losing the whole of your work. You don't even need to save a .Blend file for this to work. You can change the settings for this in the tool window at the top of the Blender screen.

Built in Hot-Key List

Since the few last versions (since 2.28 I think) blender has a built-in all inclusive hot-key list. Just open a text-editor window, and right besides the option to create a new text-buffer there is a menu option called “KEYLIST.intrr”. Select that option and the full list is loaded in memory

--Edit: There may be many hot-keys missing, especially the newest ones, like M to mirror a mesh, K for the Knife tool, **Alt+Z** for textured view... but the list is a good start at least.

Un-compiled PlugIns

Blender can load plug-ins for texturing, sequence editor, etc. . However, Blender comes with a few of such PlugIns un-compiled.

In Linux they are located on the plug-ins sub-directory of the default Blender install, and all that you need is a Make command to compile them. I don't know how to compile

them in Windows, but there they are, just waiting for you to awake them!!!

Blender “Easter Eggs” (Weird things included in blender)

All Publisher versions of Blender have been shipping with a Monkey mesh called Suzanne. Just open the main tool box (**Space Bar**) -> Add -> Mesh -> and right below the other primitives you'll see the Monkey.

Why/What it is for? Only NaN programmers know*. It is supposed to be a private joke among the blender developing team. Since then, its considered as a sort of mascot for blender. However, it is incredibly useful as a quick complex object for testing textures, shaders, etc..

By the Way, Suzanne isn't the only joke included... but I won't spoil the surprise. You will bump with them on your daily work, that is for sure.

- NaN was the company that originally developed Blender

Try this: run blender with the -Y argument (open the command prompt/terminal, go to your blender executable file folder, and type blender -Y)

Weird Error Message

Sometimes, when trying to use a boolean on a tri-based mesh, Blender gives the prompt: "Wanna Crash?" >Yes Please!

but clicking it doesn't crash. This is because Blender is a lot more stable now compared to when that 'crash' request was coded. It also used to appear when using beauty fill after face fill. (**Ctrl+F** in edit mode)

Blender 2.37 now has a 'widget', which replicates the red/blue/green axis symbol in 3DSMax in the 3d windows. Rotation scaling and movement of objects/Verts/faces etc., can be manipulated using the widget, or in the usual manner of earlier versions of Blender.

Turn your blender animation into a screen-saver (Windows)

to turn your blender animation Windows binary file into a Windows screen saver, rename the EXE file into SCR and right click it and install !

Discover the FPS rate of a Window

If you hit **Ctrl+Alt+T** key in a window, Blender will tell you how much time it takes to render a single frame of that window, in milliseconds. Valuable Benchmark

5.37.9 ???

using construction widget press **Shift** to get precision mode for fine tuning. Release left mouse button (LMB) and holding **Shift** down press it again, then you'll get moving along another axis.

5.37.10 Sculpt Mode Hotkeys

- F: an interactive brush resize
- Shift F: an interactive brush strength adjuster
- Ctrl F: in interactive texture angle adjuster for your brush.
- Shift B: a rectangular zoom selection for close-up work
- Alt B: hides all but selected rectangle
- A: toggles airbrush
- S: smooth
- D: draw
- G: grab
- L: layer
- I: inflate
- P: pinch
- V: toggles add and subtract in draw mode
- Use X, Y and Z to toggle axis mirroring.

5.38 Troubleshooting

ATI Radeon Slowdown Problems

Go the ATI site driver downloads section and select the appropriate OS/graphic card. This will (most likely) link you to the d/l for the current Catalyst 4.4 drivers.

At the bottom of that page is a link to “Previous driver versions”.

D/L the Catalyst 3.7 driver package. Run the EXE and it will extract the driver install package to:

C:\ATI\SUPPORT\

In this directory will be a directory named (for XP/Win2K users: \wxp-w2k-7-93-030812a1-010735c-efg\

This is where the driver install package is located. DO NOT RUN THE SETUP. You don't need to install the old driver.

Navigate down thru the dir structure to the following dir: C:\ATI\SUPPORT\wxp-w2k-7-93-030812a1-010735c-efg\2KXP_INF\B_10679

In this dir you'll find a file called: atioglxx.dll_ This is a "packed" version of the ATI OGL driver.

The next step specifies how you can extract these files to your hard disk using the 'expand.exe' utility included on the cd.

The 'I386' folder on the Windows XP CD contains a utility 'expand.exe' that can be used to uncompress all compressed dll files. It is a commandline utility, so you have will have to run it from either the command prompt or the Run dialog. Some examples of its usage are:

```
expand X:\I386\ADROT.DL_ C:\ADROT.DLL
```

The above command decompresses the compressed DLL

ADROT.DL_ on the WinXP CD, copies it to C:, and changes the extension to .DLL .

Just extract that file (atioglxx.dll) to your Blender install directory. Usually C:\Program Files\Blender Foundation\Blender\

Launch Blender.... no more slowdown.

Addendum: I found this solved a problem of crashing on start up after updating my Catalyst driver from 7.2 to 7.8, that is expanding the old 7.2 atioglxx.dll_ to atioglxx.dll and copying it to the Blender directory.

For people who the above solution leads to Blender crash at start up

Thanks to Xenobius at Blenderartists Forum

This solution is tested on ATI radeon express 200 (x200) integrated video card. However it should work on other ATI card.

Download the Nvidia 53.03 Driver [HERE](#)

Extract the driver to C:\NVIDIA

Copy the nvoglnt.dll from the C:\NVIDIA folder to your blender program folder (i.e. C:\Programs Files\Blender 2.40\)

Rename nvoglnt.dll (the one in the blender program folder) to atioglxx.dll

Run blender...voilà! Blender runs like it should!

[This works, because it can't use proper gl calls and uses no hardware acceleration so speed goes crappy...]

5.39 Creating Blender Libraries

You've finally made that perfect object, armature or material: a gamepiece, a robot, a fully rigged vampire character, or a millimeter-accurate model of the Empire State

Building. Besides using this work in your own artistic project, you have made the courageous decision to share that creation with the world.

Sharing your creative work is a great way to "give back to the community," even if you don't write Blender code and you can't translate the Blender documentation to Swahili.

5.39.1 How to Make a Library

So how do you make a library? In Blender, you don't need to export to a special format. In fact, you don't need to do anything special beyond saving your regular .blend file. Every .blend is a library file. Users can Append what they like from your .blend file, and ignore parts which they don't need for their own project.

This scheme has some benefits and some drawbacks.

The benefit of using .blend files as a library format is that it's super easy to include extra stuff to help the user see the objects. If the user loads the .blend file directly, it works like the pretty packaging for foods, including a quick and easy way to get a pretty "serving suggestion" rendering of the library contents. What you save in your .blend is what the user will see when they load it, including all your user interface settings, lighting types, and camera positions.

The drawback of using .blend files as a library format is that it's super easy to include *unintended* things, such as extra meshes, unused material and texture channels, and other things which the user will not find helpful. Blender doesn't save things which are no longer referenced anywhere, but it cannot read your mind if you leave a spare mesh on layer 13 which uses some abandoned Material.034.

Also, some people are not accustomed to the way that Blender saves all of the user interface settings along with the .blend file. When they load your mesh, they see your way of working. This can be instructive, but unless that's your intention, it's best to try to stick to a simple and clean user interface setup for your library files.

For best results, you need to apply some discipline to publish a clean and useful library.

5.39.2 Library Usefulness Checklist

After having used several library .blend files from various sources, I propose that anyone making libraries follow a few suggestions:

- keep to a certain object, theme or area of focus for each library file
- all the test cameras and lights prefixed with a dash; e.g., **-Camera**, **-HemiLight.001**, etc.
- any other components not intended for Appending prefixed with a dash; e.g., **-ground**

- all the test cameras and lights on the last one or two layers (**lower right layer button**)
- any composite object intended for Appending organized in a **group** to hold it together
- any groups, objects, materials, textures intended for Appending given **meaningful names**
- document your unit scheme; e.g., **1 blender unit == 1 imperial foot**, etc.
- any other layer contains test-render-ready objects or scenes
- choose rendering and world **settings which will not take an hour** to make a simple test render
- make visible upon loading the layers required for a camera, a good lighting angle, and a shared object
- make visible upon loading one small text file which lists layers and objects
- make visible upon loading any python script, **with instructions on how to start it** in a big comment
- make your licensing expectations clear: **artistic license, creative commons**, etc.
- pack the texture files and other data before saving that final .blend for publishing
- sign your work, stable email address or website url if possible

The dash prefixes for test-rendering cameras, boxes, floors and lights will help the user know at a glance what to Append and what not to Append from your library.

Here's a quick way to throw out all the stuff you really don't need, including extra meshes, materials and user interface complexity.

- save your current .blend (and make a nice backup file too)
- shut down Blender entirely
- open up Blender again, which will load the default settings
- delete the cube and cameras from the default settings
- Append all the useful parts of your library .blend file (including the *useful* test-rendering items)
- select the proper test-rendering camera for users to try out your model quickly (select, then Ctrl+KEYPAD0)
- adjust the views to ensure important things are visible and ready to render
- save the new library .blend, ready to publish

Blender saves the default settings in a file called **.bblend** on your disk. If your own preferred default settings are still too complicated for newcomers to understand, you can move that file away temporarily to get the "factory" built-in default settings which the Blender team produces as a part of each new version of the software. Move the file back again when you want to go back to your individual way of working.

5.39.3 Publishing Your Library

It's helpful to include the .txt file and/or post it separately so that people can read a summary before loading the blend file. This should include any credits, usage notes, layer explanations, and licensing information. For you Unix folks, remember to run it through **unix2dos** to enforce \r\n CRLF newlines, readable by people with less flexible tools such as Windows Notepad.

It's also of immense benefit to put up small test-renders of your library objects or materials. They don't need to be large but they should give an honest view of the work you're sharing before a potential user takes the time to download library files that will not be useful to them.

So, where do you publish your work of artistic genius?

- <http://www.e2-productions.com/bmr> Blender 3D Model Repository -- link has been taken over to serve malware
- <http://blenderartists.org/> forums
- <http://www.deviantart.com/>
- <http://www.blendswap.com>
- your own website

If you post things on your own website, try not to rely on a free site that will over-run your bandwidth limits and disappear two months later. Search engine links will sometimes live on for years, just frustrating those who were hoping to find a millimeter-accurate model of the Empire State Building.

5.39.4 Beyond Libraries

If you have even more time to spare, consider writing up a tutorial on how you achieved any tricky results!

5.39.5 Thank You

For every artist who chooses to share their creative works with the community, there are a dozen artists, or even hundreds, who thank you immensely.

5.40 Add some depth with stereo

This tutorial contains some tips for how to work with stereo images.

Stereo viewing is to see the same thing from two slightly different angles. This is what humans normally do with their two eyes. There are many ways to view a stereo image. A perfect method is to use a stereo monitor. But they are quite expensive, and many are of very low quality. In the other end of the price-scale you can use crossviewing. Put the two images next to each other, and look at one with one eye, and the other with the other eye. Needs a bit of practice though. A popular solution is red/blue anaglyph glasses, but they give very bad colors.

5.40.1 The stereo camera

I wanted a stereo camera rig that was easy to work with. It should have three cameras (center, left, and right), have an easy way to set separation (should be 1/30 of the distance) and the center cam should be used to control position etc, the two others should just follow.

Create it

Manually

1. Reset the cameras position, rotation and size (you can use Alt + G , Alt + R and Alt + S). You may want to note these values first, so you can change them back later. I just make sure there's an IPO-curve for them, for example by making a keyframe (I).
2. Create two new cameras. Reset their position, rotation and size too.
3. Name them *Camera.Right* and *Camera.Left* or similar. Set LocX to 1 for *Camera.Right* and -1 for *Camera.Left*. (Press N to see data for selected object, in this window you can change name and values.) (Select a camera by clicking RMB several times on the cameras, until the right one is selected.)
4. Create a new cube (remember to press Tab to exit edit mode). Reset position, rotation and size. Name it *Distance Cube* and set these values: LocZ: -30. SizeX: 0.1. SizeY: 0.1. SizeZ: 30.
5. Switch to front view. (1)
6. Select *Camera.Left*. Then select *Camera* while holding down Shift so both are selected. Press Ctrl + P and press Enter to register *Camera* as parent. Then press Ctrl + L and select *Camera Data*. Repeat with right *Camera.Right* and *Distance Cube*.

```
With a script import Blender from Blender
import * # Prepare scene = Scene.getCurrent() camera
= Object.Get("Camera") # TODO validate that
it is a camera oldLocation = camera.loc oldRotation
= camera.rot oldSize = camera.size # Create
stuff c = Camera.New("ortho") cameraLeft =
Object.New("Camera", "Camera.Left") cameraLeft.link(c)
scene.link(cameraLeft) c = Camera.New("ortho")
cameraRight = Object.New("Camera", "Camera.Right")
cameraRight.link(c) scene.link(cameraRight) dEmpty =
Object.New("Empty", "Distance") scene.link(dEmpty)
# Configure camera.loc = (0,0,0) camera.rot = (0,0,0)
camera.size = (1,1,1) cameraLeft.loc = (-1,0,0)
cameraLeft.rot = (0,0,0) cameraLeft.size = (1,1,1) cameraRight.loc = (1,0,0)
cameraRight.rot = (0,0,0) cameraRight.size = (1,1,1)
dEmpty.loc = (0,0,-60) dEmpty.rot = (0,0,0)
dEmpty.size = (1,1,1) scene.update(1) # Connect
camera.makeParent([cameraLeft, cameraRight, dEmpty], 0, 0) # do that CTRL+LKEY
thing cameraLeft.link(Camera.Get("Camera"))
cameraRight.link(Camera.Get("Camera")) # Reset original
values camera.loc = oldLocation camera.rot = oldRotation
camera.size = oldSize # Finish Blender.Redraw()
```

How to use it

- Never change the cube or the two side-cameras. Only change the center camera. Use that one for positioning, rotation etc.
- To set the separation: As always, select the center camera. Resize it (with S) so you can see the end of the cube if needed. Point at the end of the cube with the mouse pointer, and press S . Move the mouse pointer to the point of the main motive, that is closer to the camera. The end of the cube may not end exactly there, but that doesn't matter.
- To render (or preview) with one of the side cameras, select it and press Ctrl + O .

What needs improvement

The cube is visible Instead of using a cube, I'd rather use something that doesn't render. It could be an "empty", but see below.

The cube is hard to see It can be hard to see where the cube ends, even in a simple scene. You can select the center camera AND the cube, then it is clearly visible. But you must remember to only select the camera, before you insert a keyframe, so your changes to the cube doesn't get saved.

Distance plane is at infinity The distance plane is where "zero depth is". When viewing a stereo image, the

distance plane is where the medium is. In this rig the distance plane is at infinity, meaning *everything* is in front of it. While stuff popping out in front of the screen is cooler than stuff being “inside” the screen, it’s a lot harder to make it look nice. Specially because with everything in front of the screen, it’s easy to get stuff that is just way to close to the viewer. It can get so hard to see that the 3d-effect is completely gone.

A different solution is to make the side cameras point at the end of the cube, or add a plane to the rig, and point at that. But then the cameras are no longer parallel, and that creates distortion.

The compromise solution is to (conceptually) render the images too wide, and then crop the excess from the left side of the left image, and the right side of the right image. Then it will look like it points just like in the previously mentioned solution, but without the distortion.

But this give a lot of problems. First, I’d like to do this a smarter way, instead of just following the path describe above. But I don’t think that’s possible in Blender. (I think povray can actually do this.) So I need to render the image too wide. But if I increase width in output, it actually renders the same width of the motive, but decreases height. Then I need to do some weird math to get the right FOV, and I don’t know half the formulas. Cropping must be done outside Blender, and the rest of “the production line” is hard to get back into Blender if wanted.

I guess a couple of planes in the rig close to the cameras could simulate the cropping, but that is of limited value. To make an adjustable distance plane with them seems quite hard to me, it would need some scripting I guess.

5.40.2 Stereo viewing with the rig

This is where it gets exciting, now you are actually getting something to look at.

1. Create a new screen, you could call it *Stereo View*.
2. The screen should have one big area from side to side.
3. Set the area to *3D View*
4. Unlock it (The *Locks layers and used Camera to Scene-lock*)
5. Select *Camera.Right* and press **Ctrl + 0**
6. Split the area in half so there is two parts next to each other
7. In the *right* area, select *Camera.Left* and press **Ctrl + 0**
8. Adjust zoom in both areas so the frame that shows what is rendered is completely visible, but fill as

much as possible. If you can’t make them same size, the two areas are not same size.

9. Use cross-view method to see the 3d-effect. Look at the left area with the right eye and right area with the left eye. (It’s a skill you need to learn and practice.)
10. Press Shift + Alt + A to see your scene animated in stereo.

If you cannot see these, there is a lot of help with viewing them on the web. Use a search engine to find them.

5.40.3 Stereo rendering

Anaglyph

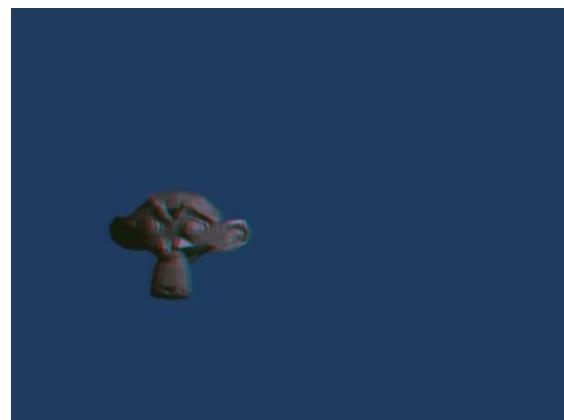
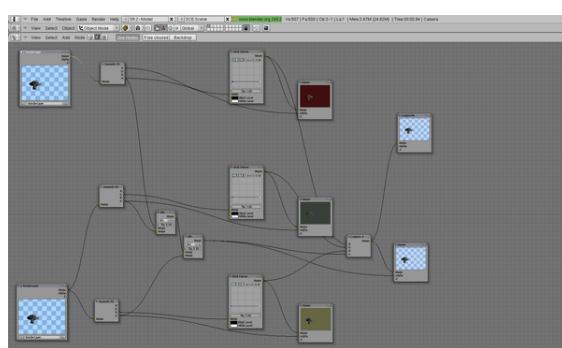
Create two render layers in the *Render Layers* tab.

Then go the *Node Editor* and click on the button with a human face. Click on *Uses nodes* and remove the created items by selecting the items and clicking on **Del**. Now, using the *Add* menu, you need to do this diagram:

Once done, return to the *Buttons window* and select the button *Do Composite* on the *Anim* tab:

Then click on the button *Render* on the *Render* tab:

You should see your objects as an anaglyph:



5.41 Ways to create a “fluffy” effect (materials and lights)

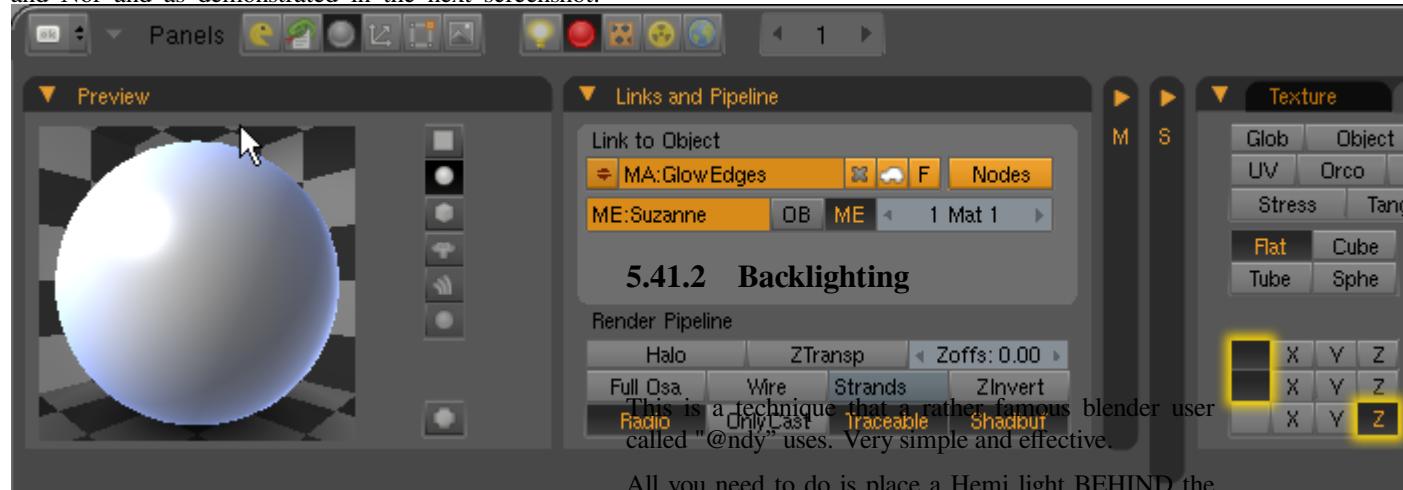
5.41.1 Spherical Blend Texture

This technique maps a spherical blend texture to the outside of an object.

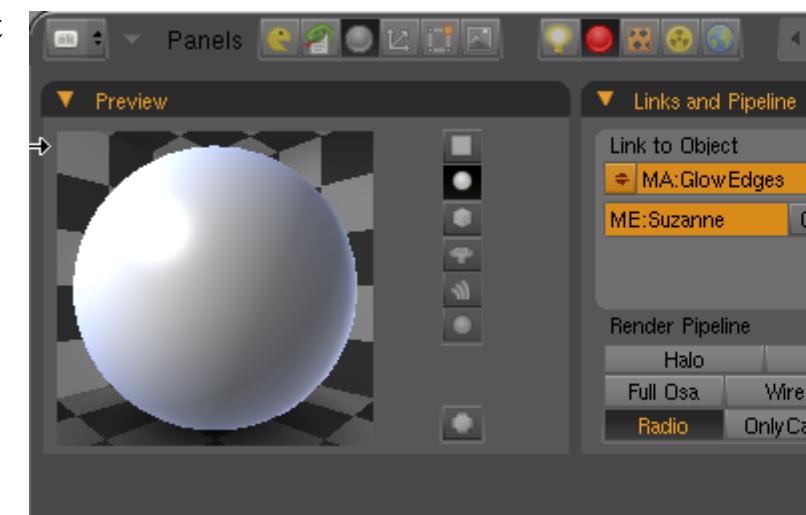
Create a new material and name it something intelligent. Create a new texture for this material, change the texture type to Blend, and then in the texture properties, change the blend shape to “Sphere”.



Back in the material buttons, go the “MapInput” tab and change the texture coordinates to Blank-Blank-Z and Nor and as demonstrated in the next screenshot:



Finally go the “MapTop” tab, and depress the Emit button and set the Texture Blending Mode to “Add”:

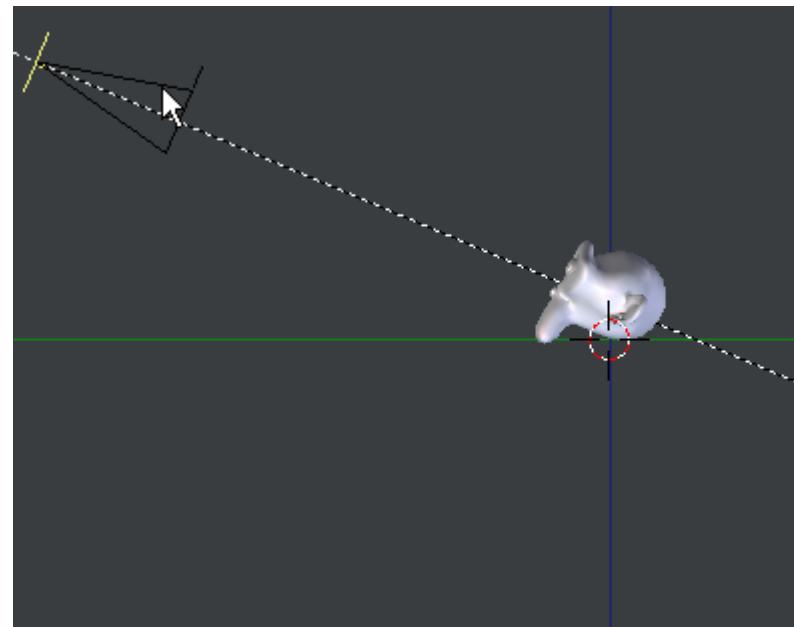
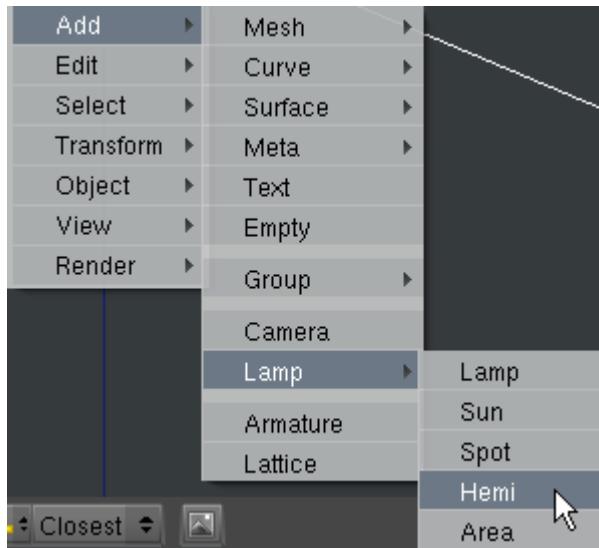


That's all there is to it, and here is how it looks:

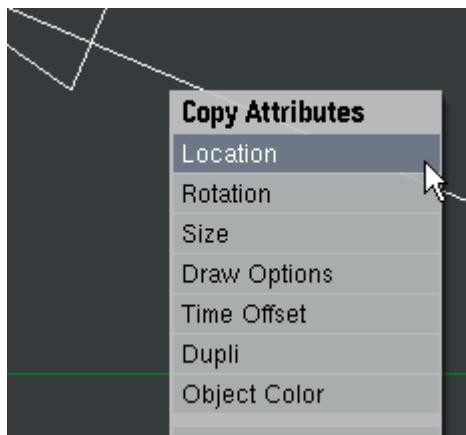


All you need to do is place a Hemi light BEHIND the object you want to light in respect to the active camera.

Breakdown: First add a Hemi lamp to your scene:



Next, Select the lamp THEN press and hold down the Shift key, and select the Camera. Press Control-C to bring up the copy attributes menu, and copy location AND rotation.



Now select JUST your lamp, press R to rotate it, then press the X key TWICE to rotate around the local X axis. Using your numpad, key in "180" to rotate the object 180Degrees.

All that remains to do is to press G to grab the lamp, then press Z TWICE to move along the local Z axis, and move the lamp until it is past and behind the object of interest. your resulting setup should look something like this:



- Color ramp with input set to normal.

Pretty straightforward, but many advise against it.

- Minnaert shader

Available in 2.37, "Darkness" <1 actually brightens edge. A cool shader, but not very useful for this purpose.

5.42 Human Body

This tutorial should cover the modelling, skinning and animation of human body, plus facial expressions.

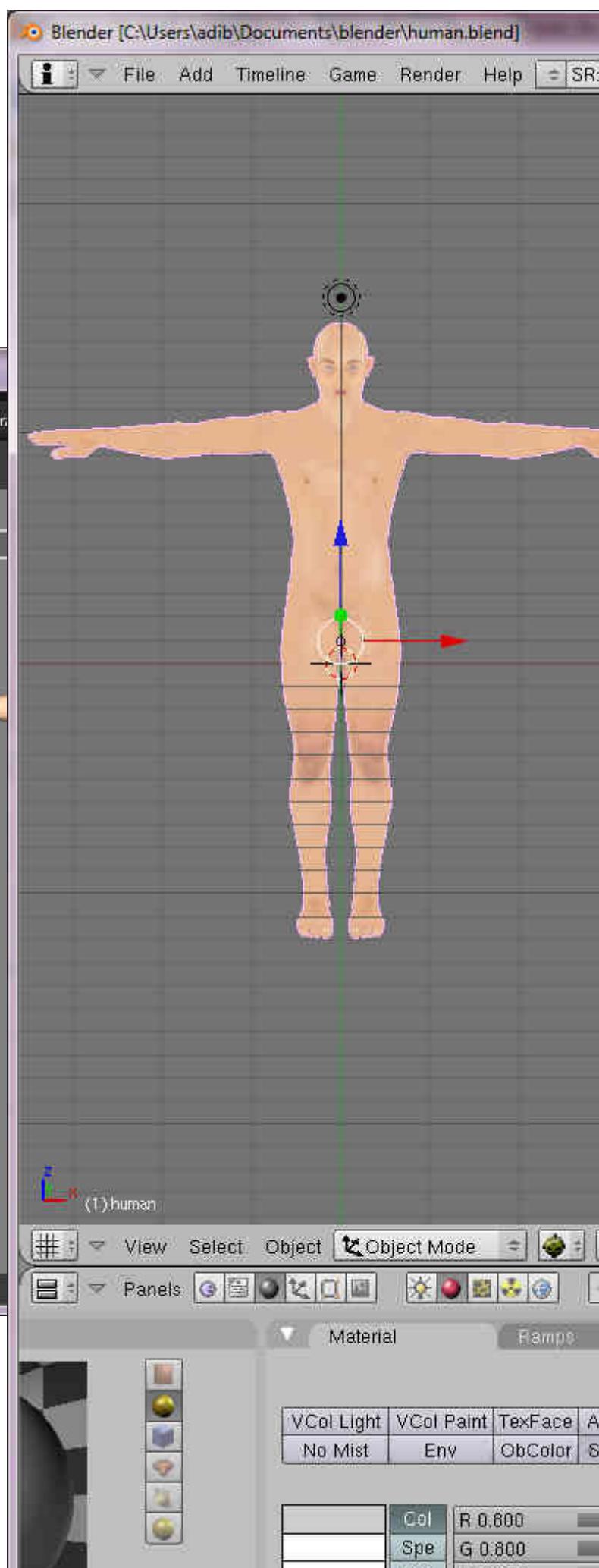
There are some tools to generate human body from parameters like:

- makehuman free software program (gpl). The models generated are under MIT license. The models generated can be easily imported in blender.

Use lightwave(obj) export. The exported file is located at
...\\Documents\\makehuman



Delete the original cube first. The human object is smaller than the standard cube. Use the UVImage editor to see the UV mapping but it actually uses a texture stored in the image file texture.png



- facegen proprietary program. Used for The Elder Scrolls IV: Oblivion.

links to anatomy sites :

- fineart
- Body proportions

tutorials: Blender 3D: Tutorial Links List

5.43 Rendering Information

Only some links to wikipedia articles for now

- wikipedia:Rendering (computer graphics)
- wikipedia:Specular reflection
- wikipedia:Diffuse reflection
- wikipedia:Flat shading
- wikipedia:Gouraud shading
- wikipedia:Phong shading
- wikipedia:Lambertian diffuse lighting model
- wikipedia:Phong reflection model

5.44 Using Blender Libraries

You can append libraries in two ways. You can make a local copy of **data blocks** (like objects, meshes, ...) of the content of a .blend file in your .blend file (**Appending**) or you can use data blocks from another .blend file (**dynamic linking**).

You can **Append** with the command in the 3d view File -> Append or **SHIFT+F1**. When you give this command a file browser window opens. There are two buttons **Append** and **Link** at the bottom of the window. The default action is *Appending*. But you can use *dynamic linking* selecting the link button.

At this point select the .blend file to append. You can select one of the following data blocks type to append:

- Group
- Mesh
- Object
- Scene
- Text

- **World**

- (not complete yet)

Note that this is a complete list, when you append only block types present in the file will appear. Select the desired type. Now you can select the particular data block to append by selecting its id.

5.44.1 Indirect linking

When you give an append or link command almost all relations between **data blocks** in Blender get expanded. For example, when you link (or append) a specific Group, all its objects, the meshes associated with the objects, the materials and the animations will be linked (or appended) too. That is called “indirect linking”. When you use *dynamic linking* such indirect linked data is not stored when you save a .blend file, when you load the file again blender will look for the indirect linked data blocks in the library file.

5.44.2 Groups

When you *append* a Group, blender will also create links in the current *Scene* to the objects that are part of the group. The Objects then become visible. However, when you decide to *dynamic link* a Group, it won't do that. To use the objects in your blender project you can use the group as a **duplicator** “**SHIFT+AKEY** -> Group menu -> group id to duplicate”.

5.44.3 See also

- http://www.blender.org/cms/Library_Linking.769.0.html
- http://www.blender.org/cms/Blender_Architecture.336.0.html

5.45 Beginning Modeling Final Project

Now that you've gotten the hang of 3D modeling, it's important to get some community feedback on your progress. Don't be an idiot and skip this part, or you'll regret it later. Basically this will help you track your progress and give you something that you'll be working on over a long term and something you'll be proud of.

- First, you need to come up with a project idea. You can choose your own modeling project, or choose one from the list below.

- Second, you need to create a model of your idea. Spend a couple of hours on it, and give it some details.
- Third, once you believe you've come far enough with the model, post it in the [Works In Progress](#) forum on BlenderArtists.org (formerly elysium.com) (you will have to create an account if you haven't already). Post several screenshots of your model from within the Blender (note: creating screenshots is outside the scope of this wikibook, though see note lower down the page). You can post whatever subject and message with your posting that you would like, or you can use this suggested subject and message:

Subject: Beginning Modeling Final Project -
<project name>

Please assist me with any feedback on my model, keeping in mind that I am an absolute beginner still. I appreciate your help.

- Wait for feedback. It usually comes very quickly. If you have any questions about feedback that you are given, don't be afraid to ask your questions in the forum.
- When you and others that have viewed your work feel that you are ready, save your model in some place you can get back to easily. You will continue working on this project once you've learned some new skills.
- Move on to the next page.

(BTW. in Windows and/or maybe other OS, to take a screenshot press 'PrtScn' (PrintScreen). It will copy the screen to clipboard for you to paste in your favourite graphics application. This may not work in other OSs but try anyway. You can also create Blender screenshot directly from Blender using menu **File>Dump 3DView...** or **File>Dump Screen...**)

(In linux under the KDE I use ksnapshot, check under the graphics tab and see if you have it. If not it should be just a google search away :) gl and happy blendering)

(On Mac OS X, press Command (Apple) + Shift + 3 to do a full screen capture)

- List of ideas:
- A Computer and keyboard
- A fishing rod
- A train engine
- A skyscraper

- A robot
- A Tank (real or made up)
- An airplane
- A truck or car
- Household appliances
- A Weapon

5.46 Using Inkscape to make advanced Bezier curves

5.46.1 Introduction to Inkscape

Inkscape is a free program that uses SVG (Scalable Vector Graphics) for its file format, and these SVG files can be imported into blender as Bezier curves. Inkscape has some great tools for making advanced shapes that would take forever to make in Blender itself. We will use Inkscape because it is free, and it is very easy to learn.

Installing Inkscape

- *Mac OS X and Windows*: Go to [Inkscape.org](#) and click download. there will be installation files for each operating system.
- *Linux*: Go into the terminal.

For Arch Linux, type **pacman -S inkscape**

For Debian based distributions, type **sudo apt-get install inkscape**

For Fedora and rpm based distributions, type **yum -y install inkscape**

For any other distribution, check your package manager to see if it has a package for inkscape. if not, download the source code at [Inkscape.org](#) and run PKGBUILD in the directory of the source.

Inkscape is a pretty large program, so it will take a while to download and install.

5.46.2 Getting Started

Because blender doesn't import any effects or filters from SVG files, we will only have to learn basic path modeling. This is very easy to learn, and will take no time at all. Once it's finished installing, open Inkscape. You will see a large empty page and a bunch of buttons to the left. Don't mind the empty page, we won't be needing it. We will resize this once we are done with the tutorial. Before we get started, I will show you how to use some of the controls. You will probably be right clicking a lot because you're

used to blender, but that's okay. Let's start out with a star to practice some of the controls. Click the Star/Polygon tool to the left, and click anywhere on the paper and drag.

Simple Controls

go back to "Object Mode" by clicking the cursor icon to the left. Now, let's try out some of the controls:

These are only a few of the controls, you will learn more controls later in this tutorial.

Editing your Bezier path

Go back to the Star/Polygon tool so we can edit the star. You will see 2 dots appear on the star. You can drag these around, and see what happens. Another thing you can do is add more points to the star, change the spike ratio, make it rounded, and make it randomized with the "CHANGE:" Dialog right above your image.



If you want to edit each node separately without symme-

try, go back to 'Select and Transform Objects' mode and click Path>Object to Path. Now you can just go to 'Edit Paths by Nodes' mode and play around with the shape a bit.

By clicking and dragging, you can move a node or curve an edge. when you make a curve, 2 nodes will show up that you can drag and edit the curve with more precision.

To add a node in the path, select an edge and press the plus button to remove a node, press the minus button, and, well, the buttons pretty much explain themselves, so you can just play around and see what you get.

After you think you know how to edit paths pretty well, let's try to make a path from scratch. Click the 'Draw



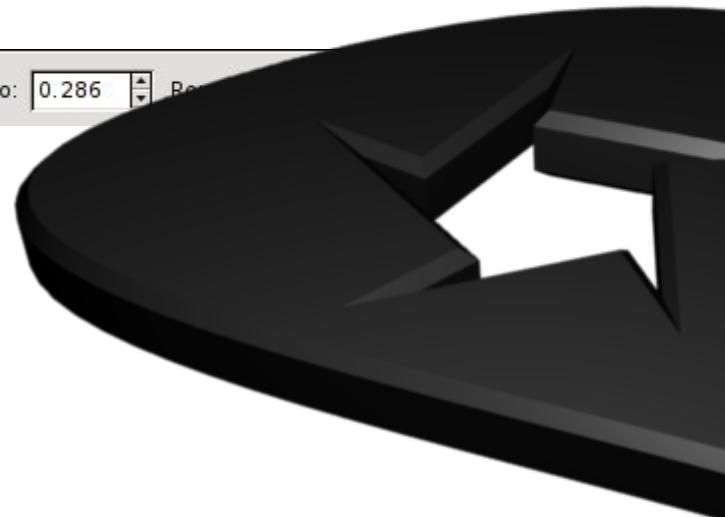
Bezier Curves' button and click to add nodes for the shape you want. If you click and drag, you will make a node and curve the node while dragging the mouse. To finish the path, go around to the first node that you made and click on it. This will finish the path and make it a solid shape. When it is done, a outline, or stroke of the shape will appear. If you want it to show as a solid shape, you can click one of the colors on the bottom and it will fill the path with that color.

Now, move the star inside the path (In 'Select and Transform Objects' mode, of course), without it overlapping the borders of the path. If the star is too big, scale it down and hold ctrl to keep the same scale on the X and Y axis. Select both your path and the star, and press Path>Difference to cut a star-shaped hole inside your

path.

Congratulations, You have made a completely pointless random shape and learned the basics of creating a shape in inkscape! Now, It's time to get this into Blender! Save your SVG image, and open up blender. Press File>Import>Paths(.svg, .ps, .eps, .ai, gimp) (Blender 2.49). Select your SVG file and the settings are fine how they are. Press OK and see your path appear in blender (You may have to delete the default cube to see it) From here, you can Extrude, bevel, and do whatever you want to you bezier path in blender.

The result may look something like this:



5.47 Light Mapping

5.47.1 Needs Expert

The original writer of this section is not knowledgeable about the subject of “Light Mapping” and hopefully can benefit from someone who has much more experience with it

5.48 Platonic Solids

The **Platonic solids** or **Platonic polyhedra** are the convex polyhedra where all faces are copies of the same regular polygon, and the same number of edges meet at every vertex. There are five of these shapes: the tetrahedron (like a pyramid but with a triangular base): cube, octahedron, dodecahedron and icosahedron.

Recent versions of Blender include an addon called “Regular Solids”, which lets you instantly generate these and a whole bunch of other similar shapes. However, the following steps do not require any addons.

The Tetrahedron

Bring up the Add Mesh menu (Shift+A), and select a Cone. Set the number of Vertices to 3, leave Radius 1 at its default value of 1.000 and Radius 2 at 0.000. Now, set the Depth to $\sqrt{2} \approx 1.414$. To make sure that you have a regular tetrahedron, you can check the lengths of the edges (in Edit Mode, press N to open the Properties panel and locate the checkbox **Length** in the section **Edge Info**).

The Cube

This happens to be a built-in shape in Blender. Just bring up the Add Mesh menu, and select Cube. Done!

The Octahedron

This shape is the *dual* of the cube—it has vertices where the cube has faces, and faces where the cube has vertices. To make it, first create a cube. Press Tab to switch to Edit mode. All the vertices should already be selected. Press W to bring up the Specials menu, and select the Bevel function (or select it directly with CTRL + B). As you move the mouse, you will see each vertex of the cube turn into a triangular face; don’t bother getting the shape exactly right, simply press LMB to finish the drag. Then, look in the panel that should have appeared at the bottom of the Toolshelf on the left of the 3D view (press T to toggle its visibility); you should see an editable numeric field labelled “Offset”. Type the value 1.0 into this field, and that should exactly form the octahedron shape.

Finally, bring up the Specials menu again, and this time select Remove Doubles.

The Icosahedron

Bring up the Add Mesh menu, and select an Icosphere. Set the Subdivision to 1. Simple!

The Dodecahedron

This shape is the dual of the icosahedron. To create it, make an icosahedron as above. Then do what you did to make an octahedron out of a cube: press Tab to switch to Edit mode. All the vertices should already be selected. Press W to bring up the Specials menu, and select the Bevel function. As with the octahedron, press LMB to finish the drag. Then set the Offset value to 0.607, the closest I could find.

Then, bring up the Specials menu again, and this time select Remove Doubles. However, the default Merge Distance of 0.0001 will not be enough to remove all the doubled vertices; change this to 0.001, and you should see the message “Removed 40 vertices” briefly flash up.



Exercise

What’s the dual of the tetrahedron? Try applying the Bevel operation to one of those; what do you end up with?

5.48.1 External Link

[Wikipedia article](#)

5.49 Polygonal Modeling

As the name suggest, polygonal modeling involves constructing the models out of polygons. Strictly, software packages works only with triangles. Blender could show you quads and some software packages could even let you work with polygons with more than 4 sides (n-gons) but these packages work internally only with triangles. These triangles are hidden so that you could concentrate more on modeling.

While different modelers works differently, they would still follow certain workflows (some call this techniques). These are familiarly called box modeling and polygon-by-polygon modeling (poly-by-poly modeling).

Box modeling is a top-down approach where modelers start with a primitive (usually a cube (box), hence the box in box modeling). Then from this primitive, the model's form is build up where details are gradually added.

Poly-by-poly modeling is almost the opposite of box modeling. It is the bottom-up approach where modelers start with a plane or even a vertex. Using extrusion and other tools, more geometries are added. Polygon by polygon the model is gradually build.

Neither of the two is better than the other but there are instances where one workflow lends much better to the situation. One typical advantage of box modeling is that at early stage, you are able to conceive first the form (the whole) of the model and is better equipped to do general corrections without bothering yet with the details. Poly-by-poly otherwise give you more control on the geometry and is much more easier to use in modeling complex forms like the human ear. Complications do arise when working with the two. In box modeling, it is typically difficult to manage the geometry when adding details. Experience is necessary so as not to create a mess. Poly-by-poly modeling otherwise has its own quirks. Since you are going detail by detail, it is easy to make a mess with the models form (i.e. wrong proportions).

While the two are opposites, they compliment each other beautifully. Most modelers would combine the two workflows in a variety of proportions, combining the strength of each. For example one might model a human body using box modeling while the head and the ears are done using poly-by-poly modeling.

Blocking with primitives is added to enrich approaches in modeling. It is basically box modeling in conjunction with divide-and-conquer approach. Different parts are modeled separately with their own primitives and later attached with the others to form the whole.

5.50 Box Modeling

Easy to undertake and flexible, box modeling is a favorite of beginners and veterans alike. It is fun to work with and general results are visible in a short time. It is a powerful work flow that any modelers should know.

5.50.1 Before We Start

Before doing any modeling, it is vital to plan first. Gather references and make a general plan on how to tackle the modeling phase. Being ready will save you a lot of complications later

5.50.2 The Work Flow

With box modeling, start with a primitive that is appropriate for your subject. Most start with a box and generally a good primitive to start. Then, from the primitive, using a variety of tools, mold the essential form of your model. Also called the roughing phase. Don't delve on details here, those are to be tackled later. Finally, on this form, we go to the nearly recursive process of adding details. Layer by layer, details are added until the required amount of detail is achieved.

Start with a Primitive Select the proper primitive for your work. Most modelers start with a cube. The cube is the most flexible primitive available and is suitable to almost all form of subjects. But, in many cases, selecting other form of primitives would cut out most of the modeling required. A gun is a cinch to model using a cylinder. With a little adjustment, a torus forms a good doughnut. In an instant a ball can be made out of a sphere. So produce your primitive to work with.

Rough It Out Take the primitive and start modifying it to capture the essential form of your subject. Tools like extrude, loop cut, scale and grab are very handy for this. Avoid worrying about details at this stage. Those are to be done later.

Adding Details, Details and more Details Now, we approach the most difficult and unquestionably the most fun part. Using a variety of modeling tools, modify the form again to incorporate details. Add details layer by layer, from general to specific. Planning, studying and experience would help you go through. Continue until you achieve the desired level of detail.

Adding details requires subdivision or addition of polygons. The knife and the loop cut tool is handy for subdividing meshes. Extrusion is great for adding details like horns and fingers. Face and edge loop must be herded to other directions at times and warrants a study by itself. Blender provides you with required tools to achieve all of this.

At this stage, beginners and experienced modelers alike will find their meshes getting more and more difficult to work with as the mesh gets a more and more dense at each level of details added. Planning, an eye to the edge flows and experience alleviate this. So plan and keep practicing.

5.51 Illustrative example: Model a Chair (Swan Chair)

Now let's start with real modeling. In this tutorial we will study in greater depth how to model using box modeling

work flow. Let start with a swan chair. It is great piece to model for its a popular piece of furniture so reference images are easy to find. It also have a simple geometric structure so its shape easy to grasp and also contains simple edge and face loop flow so as not to necessitate difficult edge loop face loop flow herding techniques.

The techniques learned could also be used to model other chairs (Example: the Egg chair, Tulip chair or Parson's chair which is advised to be modeled too) and other subjects.

5.52 Model a Chair-Preparations

Many people do not know what a Swan chair look like. Its time to open up the internet browser and search for images. Or if you have an actual Swan chair with you, your lucky. You can look at a real model while modeling. This would simplify decisions and remove the need for guessing certain details that you are not available with image searches alone.

5.52.1 Knowing our model

We need to know what we are modeling look like. You can't do this without references to guide you. A simple image search with Google would provide you the references you need.

Gather several images that gives you different views of the chair. A front and a side view is great to use as a background image to guide as while modeling. Also, gather at least one image that shows the chair at an oblique angle so that it will provide a much more "3d" view. This will shows more clearly the shape of the chair, more than the front and side view alone would give you.

5.52.2 Loop study

Take time to study the model. For simple models like this, it would be quite quick, especially with experience. For this project, the loops are as shown bellow.

The pink loop goes around the edge of the chair defining the chair's edges and thickness. The light green, light blue and light purple loops follows and create the form of the chair. The crossing green loops forms the bottom of the chair. The light blue and green loops contribute for the form of the "wings", while the purple and green loops form the back. Its a simple loop structure, its topologically (sorry for using the term) the same as this cube extruded to form a T (upside down so the loops would match more).

While modeling keep these loops in mind and try to achieve the loops. For this project, the loops are achieved with simple extrusions of a cube, just like the T piece above, but in a form that looks more like the chair.

5.53 Model a Chair-The Seat

After studying the subject, we are ready to tackle modeling it. Start up Blender.

5.53.1 Start with a Primitive

For this project we would use a cube. The cube already has several face loop we need and none of that we don't, so this primitive is a good starting point.

Opening Blender, we are already supplied with the default cube which is handy. If you don't have the default cube because you change the setting or remove it, simply add a new cube. Name this primitive something sensible like "Swan Chair" because it would become our chair later. Move the cube or adjust the background image until the cube is centered at our image. After doing that we proceed shaping our cube to create the basic form of our chair.

5.53.2 Rough it out

Since we are modeling a symmetrical object, lets take advantage of it and use the mirror modifier. Cut the cube in the front exactly in half by using loop cut at center (Shift + RKey -> middle click, or Ctrl + RKey -> middle click on this editor's computer). For those with no middle buttons you could use subdivided. Select the vertical edges (go to edge select mode first, Ctrl + Tab -> 2) then use subdivide tool (WKey -> 1). Remove the left half by deleting the vertices. Then add a mirror modifier in the modifier stack. This is available at the edit buttons, select the modifiers tab the click the "Add Modifier" button and select "Mirror" from the selection. Activate "do clipping" from the mirror modifier panel that appeared to prevent accidental movement of the central vertices.

(Noob Note: If you follow these instructions, you'll have a gap between the mesh and its mirror. I fixed this by increasing the "Merge Limit" option in the Mirror modifier menu to .1.)

Move the vertices so that it would follow the shape of the bottom of the chair image.

Note that the reference images used does not match very well. This is because of perspective and camera angle distortion. In this case use the side image since it shows less distortion. Use the front reference image as an eyeballing guide for the front side of the chair. An oblique reference as stated earlier in planning stages is helpful to guide you in this condition.

Add an edgeloop using the loop cut tool (Shift+RKey) as shown in the image bellow. Move the newly created vertices to follow the references

Its is good practice that while modeling you had to view your model at various views not only the standard front,

side and top views. Remember that you are modeling a 3d object. What looks good in this some views are not necessarily good in other views. This would result in complications like the familiar flat face look in face models. Moving your view around would help you prevent this. Shown bellow is the view of the model at an oblique view.

Now extrude the back part of the model to form the back part of the chair. Adjust the newly formed vertices to follow your guide images as shown bellow.

Its a good practice to see the model at various views while modeling here is how it look like now at an oblique view:

Select the following faces. We are going to extrude it to form the sides for the chair.

Extrude the faces and adjust newly formed vertices to conform with the guide images.

And now the oblique view.

The back still does not look good so add the following edge loop.

Adjust it.

Remember to rotate your view to see problem areas.

Add this edge loop too.

Adjust.

And keeping with our good practice.

Say, the form is finished. The loops are all in place and the form is easily identified as a rough model of the Swan chair.

5.53.3 Adding More Details

Now the basic form of the model is finished and with all important loops in place detailing this model is easy. Lets start by defining the back. Add these edge loops.

Move the newly created vertices to fit the references.

Be sure that the model looks right at various angles. The references images collected earlier will help. Adjust vertices if necessary.

The sides need to be taken care of too so add these edge loops.

Then adjust the vertices.

This is how the model now looks like.

Continue by adding these loops.

And adjust as usual.

Check that it looks good at other views too. Move vertices as necessary.

Keep repeating this procedure. Add loops then adjust. Here is the loops that was added to the model. Tweaked after adding each.

The outline of the chair needs rounding up. Add these

loops and tweak so it looks like the one shown.

This is now how the mode looks like.

Its rather tedious to add more loops at this time so lets do a cheat. Select all vertices (press AKey a few times) and then subdivide (WKey -> 1Key) to add more vertices.

The model looks blocky so lets smooth it out. While the whole chair is still selected smooth the vertices by clicking Smooth a few times (Edit buttons ->Mesh tools -> Smooth; also available at WKey->Smooth). It would smooth out the model.

Admire the work (and tweak if necessary).

This is optional. Add this loop and scale along normal (Alt + Skey) to scale it in. Adjust vertices if necessary.

5.54 Model a Chair-The Feet

Now the seat is finished, time for the legs for it to stand on.

5.54.1 Starting Primitive

The leg is mainly made up of cylinders so a cylinder is a very good primitive to start. In object mode add a cylinder with 16 vertices.Center the cylinder on the upper part of the telescope.

Tip: to make the cylinder centered relative to the seats center, while in object mode select the seat then hit Shift+Skey (Snap options) then select Cursor->Selection. The 3d cursor would be moved to the center of our seat. Now when you add the cylinder, the cylinders center would be located at the cursor which is also at the seats center. Now all you have to do is to move the cylinder along the y and z axis because it would be properly located relative to the x axis.

Hide or move the Seat to another layer so that it would not intrude while modeling the leg.

5.54.2 The Upper Telescope

Go to edit mode and scale the cylinder until its radius match the radius of the upper telescopic element of the chair's leg. Then move the top and bottom vertices to match the reference.

Extrude the bottom part and place it a little lower. Extrude again and right click so that the newly created vertices would remain in their position. Scale the vertices a little. Extrude again and move the extruded region up the cylinder.

Remove the top and bottom vertices that cap the ends.

We will be using subsurf modifier to make this part smoother. Add this loop to constrain the smoothing on

this area.

5.54.3 The Lower Telescope

Now lets start with the lower part. Add a cylinder in object mode and adjust the newly created vertices to match the reference.

The legs would be extruded from this cylinder. Add this loop to prepare.

Select the faces between the newly added loop and the bottom loop. Extrude this faces but right click after extrusion so the extruded faces remain in their position. Now scale them constrained along the xy-plane (Skey->Shift+Zkey) a little. This is how it should now look.

In top view select the two topmost faces and extrude to form the back leg. Shape the tip according based on the reference. Here is the steps taken to shape the tip. First after extruding, Constrained scale the tip along the x-axis (SKey->XKey) to make the tip thiner. Then move the tip lower. Deselect all, then select the top vertices of the tip and lower this vertices further.

Shape the base as shown bellow. This is done by selecting the vertices where the leg connects to the base then constrained scaling it along the y-axis to zero (Skey->YKey->0key). Then move the scaled vertices further back.

Then add an edge loop as shown bellow. constrained scale the newly created edgeloop along the x-axis to make it thiner.

We will be using the subsurf modifier to smooth our model so lets add the following edge loops so that the model will smooth properly. Add an edgeloop at the tip.

And this loops on the side.

Also remove the vertex that cap the top because we will not be needing it and it would cause smoothing artifact when subsurfed.

Now how about the other legs? We will be using Spin duplicate tool to make those. In top view select the central vertex then snap our 3d cursor to it.

Now remove the other part of the cylinder as shown. We would be left with a quarter part of the whole leg assembly.

Select all the vertices that are left. Change the parameters of the spin tool to this: degrees to 360, steps to 4. Click on the “Spin Dup”. If you have multiple 3d views, your mouse cursor would turn into a question mark. Move your cursor to the top view and then click. The other legs would appear as shown bellow. This legs are made of separate mesh and the original leg have a duplicate at exactly the same position creating double vertices. Connect the mesh and remove the doubles by selecting the whole mesh then remove the doubles (Wkey->Remove Doubles).

5.54.4 The Stops

Now lets create the Black Stops on every tip of the leg. We would be creating this joined to the legs as one object then separate it later. Its just that it seems more easy that way.

Without leaving edit mode, add a cube. scale an move it to one of the tip. Move the cube so that it would be lower than usual to separate it from the other mesh making it easier to edit it without disturbing the others.

Shape the cube to match the tip. This should be easy by now. The whole step is left out.

Subsurf would be used to this too so add extra loops to control how the mesh would be smooth.

Now make copies of the stops using the Spin Duplicate tool. Don't forget to remove the doubles.

Select all the tips then move them together up until they intersect with the tips of the legs.

Now lets separate separate meshes to individual objects. Press Pkey and select “all loose part” option.

Notice that the other part of the mesh would suddenly change in appearance. Don't worry. It just means that those meshes are now separate objects by them selves. If you go to object view you would now be able to select the leg and the tips as separate objects.

There are some issues though when you use the subsurf modifier on the leg base as shown.

Select the leg and go to edit mode. Add the following edge loop.

Scale the base loop too as shown.

The issue should now be solved.

This is how the whole leg assembly should look. All have subsurf modifier on.

There is still an issue though. This is not important in most part but will be taken care of for this time.

When one of the leg stops is selected in object mode, it would show that its center is not at the center of the mesh. The center is where the gizmo (the red, blue and green arrow) would be located.

To correct this select the offending object. Click the Mesh menu (located at the header) and navigate to “Transform” and select “Center New” from the options. Blender would automatically calculate the new center based on the objects mesh. Now when the object is selected, the center would be located at a much more appropriate location.

Do the same to the other stoppers.

Now if you haven't named each individual part yet, You can do it now. Name each part sensible names like upertelescope, lowertelescope, etc.

5.54.5 The Whole Chair

Unhide or Show the layer where your chair seat is so that it would unite with the legs. Now this is how the whole model should look like.

5.55 Illustrative Example: Modeling a Simple Human Character

We will be modeling this very simple human figure. This time many details are left out to save time and space. It's assumed that enough knowledge is gained from previous tutorials for readers to understand this stage. Shortcut keys are also left out. Read first the chair modeling tutorial if this comes out too difficult to read.

5.56 Modeling a Human Character - Preparations

Actually, this model is made without any background reference images but enough knowledge of human figure makes this possible. Modeling the human body or any model in that case requires knowledge of it beforehand. Even with the background image references, there are still some details (like the armpit) that is not easily visible or are difficult to comprehend with just simple orthogonal projections. It is advised to make some study on basic human figure and proportions first.

For those who likes to have a background reference image to use while modeling, use this orthogonal views of the model. You could still try and test your knowledge on the human figure later by trying to model one without image references.

5.57 Modeling a Human Character - Modeling

5.58 Blocking the Figure

As with many models done in box modeling, let's start with a cube. The model is symmetrical so let's make use of it. Add a loop running vertically at the middle and delete the vertices on the left, leaving you with only half of the cube. Now add a mirror modifier.

Now add a loop running horizontally and fashion out the torso.

Add a vertical loop on the torso so that we will have more geometry to work on.

Extrude the head and the legs. Add loops around the head and the bottom of the legs and extrude out the head and feet.

Now add a loop around the torso and extrude out the arms. This is all for blocking

5.59 Detailing and Finishing

Add a loop that goes through the torso and the arms and another at the sides. This geometry will enable us to round and shape the figure.

Extrude out to form a rough hand. Add an extra loop around the hand and extrude out the thumb.

Add more loops around the arm, thumb and hand. Work out this additional geometry to shape the parts.

Add this loop to add more geometry on the legs and to the torso.

And then shape.

Add more loops on the leg and shape out the form.

Now time to shape the foot add more geometry and shape out the foot.

Now take a look at your model. Adjust the form until satisfied. Work finished.

5.60 Polygon by Polygon modeling

Although quite demanding, many modelers savor the control poly-by-poly modeling allows. However, its demanding nature makes many modelers, beginners and experts alike, to avoid it. Still, poly-by-poly modeling, if properly used, is a powerful tool.

5.60.1 Before we start

For this work flow, being ready is of great importance. Have several references of your subject, especially a front and side view. Also, have minutes of study on the subject and try to make out an approach in modeling it. Being ready is crucial for this.

5.60.2 The Workflow

Poly-by-poly modeling provides the most free modeling workflow ever. This freedom, however, causes confusion on how to approach modeling the subject. Searching the internet for tutorials on modeling the face using this method, you will find hundreds and each will be different from each other. Some start with the eyes, others with the mouth, others the ridge of the nose. And then each will proceed differently from that. No wonder its confusing.

Even though poly-by-poly modeling is such a free form method and constitutes a very vogue way to start and continue, a general guideline is provided in the hope that it will help modelers in using this approach.

To Start

Poly-by-poly modeling rather sets you free on choosing how and where to start. The only requirement is that you have a geometry, a vertex or a plane to start with. Many beginners are stumped by this. Somehow such liberty leaves them undecided.

A very good approach is to start by creating the important loops first. For example, in modeling the face, you could start by creating the eye, mouth and face loops. With the loops in place, it is easy to build other geometries around them.

Another good approach is to start with the most important part of the subject. For the face, this is the eyes and the mouth. You could model these parts first and then proceed adding geometry from there.

As said, poly-by-poly modeling lets you free, so if you would like to start on other parts first, say the nose, then you could do so.

To Continue

Continuing from your beginnings is not as easy as connecting-the-dots as it first appears to be. Rather than proceeding in a connect the dots fashion, proceed by thinking of edge and face loops of your model first then creating and adjusting geometry to fit them. Such view would ease decision on placing your geometry.

To Finish

When you find that you don't need to add more geometry, you can call it quits. But still, its a good practice to take another look of your model and make trial renders of it. There might be an awry edge loop, geometries that mess the model, or triangles that you wanted out. Do some fixing and adjusting and when you finally have the model good, give yourself a pat on the back, OMG ITS A FACE YOU DID IT YAYAYAYA

5.61 Animation Notes and FAQ

5.61.1 Blender 3D: Animation Notes and FAQ

5.61.2 IMPORTANT

This page is under heavy construction. It will probably not be edited better until early May 2012. If someone sends me information on how to convert a OpenOffice odt file to a wikidoc format, then it may be done sooner. Until then this will be quite a work in progress.

5.61.3 Authors & Contributors

See [here](#).

5.61.4 Introduction

I found it quite cumbersome to find all of the little quirks, problems, and tricks in multiple sources whenever I would forget something. Therefore, I compiled all of the things I found important—primarily with character animation. Almost all the information here is consolidated from many sources. Some sources still need to be cited and any help on this would be much appreciated. For that reason I did not intend to make this public but I think there is enough need for this type of information so here it is.

Just as a special aside, I made these notes to help myself with my animation project. Although I have organized them and posted them online for my and others benefit, I will not be dedicating any large amount of time to this website. All of my extra time needs to be allocated to the animation project. However, I do not mind suggestions and am willing to possible use them. I want to make sure that the viewer understands my time is very limited.

There may be many typos or confusing areas of organization. I tried to organize the mess of notes as best as possible but I am sure there is an even better way. Feel free to fix any aspect of this.

Any suggestions or comments feel free to e-mail me at [wiki \[at\] pagodaproductions.com](mailto:wiki[at]pagodaproductions.com).

5.61.5 Todo List

See [here](#).

5.61.6 Animation FAQ

5.61.7 Armatures

Q: I managed to create an acceptable walk cycle finally, now when I reopen the file to add additional animation, It keeps falling back to walk cycle, is their something i am doing wrong. Even after adding the LocRotScale key frame, it still ignores them, and goes back the corisponding frame for the walk. Ugggghhhh!!!

Q: What is the best approach for rigging a character in Blender?

Q: How do I attach a separate object to a bone?

Q: When you try to scale the root bone of the armature(which scales the mesh), the bones move out of the natural placement causing unwanted positioning and distortion. Is there any way around this?

Q: What is a pole vector?

Q: How do I know that my armature is correctly positioned in global space in edit, object, and pose mode with proper bone rolls and axis pointing in proper directions? What are the best methods for building? And what axis is up Y or Z?

5.61.8 Hotkeys

Ctrl-C/Ctrl-V Trick One simple way to get the right name is to select the Lattice, go to F9, Link and Materials panel and where it says Ob:Lattice or Ob:Lattice.001 etc., move the mouse over this field and press Ctrl-C (don't click on it, just hover over it). This copies the Lattice name. Then select your object, go to the Lattice Modifier panel, hover the mouse over the Ob: field and press Ctrl-V to paste the name in. Now it should stay there and your Lattice should work in Edit Mode.

H hides bones

Alt+H shows all bones

Shift+H hides everything except the chain

Ctrl+I [Pose Mode] will automatically add an IK constraint

M [Pose Mode] to move a bone to a bone-layer, select it in POSE mode then press "m" and click on the desired layer button. When the bone is selected, you can see which bone layer it is on in the Editing buttons / Selected Bones panel. Whichever of the 16 layer buttons is enabled is the layer the bone will be on.

Shift+M [Pose Mode] You can use the hotkey SH-M to display a floating bone-layer panel in the 3d viewport. Select the armature, change to POSE mode, then change to the Buttons/Editing panel (F9). Under the Armature-Panel, Display Options SH-LMB the 8th button under the Display Options Bone-Layer buttons to enable it. Check out [this](#) by BlenderArtists.com.

Alt-N normalize the bone orientation This can fix problems sometimes.

Alt+P clears parent

Alt+S [Pose Mode] in edit mode allows b-bones shape to scale visual changes without affecting the bones.

W [Edit Mode] subdivides bones

W [Pose Mode] will calculate paths to see how a bones movement is mapped out along a path or If you've named

your bones correctly (with .L and .R at the end of the name), you can flip the mirrored bone name by pressing "W" and "flip name". A relevant post on BlenderArtists.com is available [here](#).

5.61.9 Approaches, Techniques, Tips and Tricks

Constraining IK or FK In the Armature Bones Panel is where you can limit the movement of the bone for IK and Limit Rotation Constraint for FK

Customizing Frame Skipping Steps(F10 playback) allows for a customizable way of skipping frame in increments.

Copying Constraints in pose mode 1. Select all bone(s) you wish to copy the constraints to, from a particular bone 2. Now, select the bone you wish to copy constraints from 3. Ctrl C --> Constraints (All)

You can do CTRL-CKEY to copy stuff from a bone to bones. The options are location, rotation, scale and constraint. Constraint is very handy when you wan to copy a constraint to other bone. The way it works is easy. The WKEY menu get some neat options too: Select constraint target: Will select the target of the bone's constraint currently selected. Flip name: Yep, you can flip name in Posemode too. Calculate/Clear path: This is a visual way to see the action linked to your armature. You can select just some bones and ask Blender to show you the paths of the bones.

It's possible to copy constraints from one object/bone to a bunch of objects/bones. A useful thing to know when doing a repetitive task like rigging all the fingers of a hand. Just select all bones/objects that you want to give a copy of the constraint, and then select the bone/object containing the constraints. Press CTRL-CKEY in 3DView, and select Object Constraints from the popup menu. The idea behind this is to copy the constraints of the active object to the selection.

Rigging Q: What is the best approach for rigging a character in Blender?

A: Apollux Honestly, I don't think that there is a golden set-up that will solve all your problems. For example, when I design a character rig I need to know beforehand what type of actions he will perform.

If you want general pointers, I guess this would do: Always set your rigs in a half relaxed fashion.. totally relax your hand for a few minutes and see the position of the palm and fingers, that is position you should model and rig-them on the 3D world. Same goes for mouth, legs, arms, etc. etc. etc.

When in doubt over IK or FK always choose IK, since it is quite easy to switch a IK rig into FK when needed and then back to IK.

Bone's axis orientation can't be an afterthought!! It is no

coincidence that every self-respecting rigging tutorial for Blender mention it... And don't trust blindly on the automatic axis orientation fix command (Ctrl-N), since it can introduce some odd solutions on complex armatures. Always check each bone by hand, even if Blender says they are fixed.

Ultimate rig design is not the in-and-all of character animation. Sometimes I would spent crazy amounts of time preparing a rig for a special sequence, and in the end realize that if I had used a simple FK rig tricked by hand during animation I would get the same results in half of the time and half the frustration. Not because it can be done means that it must be done. [BlenderArtists post here](#).

Item Pickup A good use of it is to ask a character to pick up something. By having a bone or empty for each side of the relationship (hand <-> glass), as the hand approaches the glass, you can align the two empties and fire the constraint up (1.00) to stick them together. You add another child-bone in the middle of the hand to tell where the glass will be. Thus moving the hand will move the glass. On the side of the glass just add an empty and make it parent of the glass. Add a copy location to the empty pointing to the bone in the hand we just did. There you go. Of course when the hand rotates the glass will not. For that you will need to add a Copy Rotation Constraint. Before Blender 2.40, the above method was a good way of faking parent relationship without rotation. But now we have the hinge option which does the same.

5.61.10 Problems and Questions

Separate Object/Bone Attachment Q: How do I attach a separate object to a bone?

A: To do that simply Ctrl-Tab into Pose Mode with the armature selected then select the object/mesh you want to parent to a bone then shift-select the bone you want to be the parent and Ctrl-P>"MakeParent to">Bone .

Armature Scaling Problems Q: I have been trying to scale my character which has many similarities to the mancandy and ludwig rigs. However, when you try to scale the root bone of the armature(which scales the mesh), the bones move out of the natural placement causing unwanted positioning and distortion. Both the mancandy and ludwig rigs have this same problem. Is there any way around this?

A: Why are you trying to scale the root bone (to scale the whole model)? Are you doing this in pose mode or object mode? If you are trying to scale the whole model try it in Object Mode. Also check if you are scaling around the cursor instead of median point

Q: What is a pole vector?

A: Vertex Pusher 2) Pole vector is actually a term used in Maya. It refers to the direction a bone is set to point towards in Maya rigging. For example, the direction you

want the knees to point towards can be set using pole vectors. Apparently, Blender should be getting this feature soon . Right now you have to add target bones so that the knees will only bend one way (see Ryan Dale's BSoD tutorial), but with the option to use pole vectors you won't need to do this in the future .

[BlenderArtists post about bones](#).

Bone Orientation Q: How do I know that my armature is correctly positioned in global space in edit, object, and pose mode with proper bone rolls and axis pointing in proper directions? What are the best methods for building? And what axis is up Y or Z?

- A: 1. Build up all the bones, as you did before. (Every foot, arm, etc. should be a bit bent, and drawn from the proper view.)
2. Select all these armatures, and with Ctrl-J join them into one armature. (Or with parent/child connection, as is necessary.)
3. Select this armature, go to "Editing" (F9), and then turn on "draw axes". Then you can see the axes of the bones. You will see, that the axes are different. Some bones has the Z-axis up, some has the Z-axis to the left, etc.
4. First you should tell Blender that this is the original position of your skeleton/model by pushing Ctrl-A. This is applying rotation. This means, that now all the rotations of the bones will be "0, 0, 0", so if you push later Alt-R (clear rotation), you will get this position. (Turn on numerical menu with N, to see what happens when you make this apply rotation command.)

5. Select the whole skeleton/model again. Go to edit mode. Then push A, to select all the bones. (They should be yellow now.) Then hit Ctrl-N, and you will see all the axes will be changed to stay on the same way.

6. That is, your skeleton now is ready to define the IK solvers, and then the skinning, and finally the animation.

<http://blenderartists.org/forum/showthread.php?t=25653>

A: Vertex Pusher 4) Yes you should always add in top view[numpad 7]. An armature bone always has its axes set (Y is always up X side to side and Z depth) and you cannot change this like you can with mesh objects, so it seemingly won't matter which view you add a bone while in Edit Mode but in Object Mode your armature object will have a 90 degree rotation around the X axis (if you add it in front view) which can and usually does create problems down the road . If for some reason you want to add bones in front view simply start by adding the first armature bone in top view ad then switch to front view to add additional bones . You can just delete the first bone or rotate it while in Edit Mode .

Blender's default views have the Z as the "up" axis while the armature system is coded with the Y as the "up" (like

most 3D apps)

When you rotate a bone in the front view, when the bone is aligned to that view and the Z is the up axis, you will give a roll value the bone because it is rotating perpendicular to its coded up axis ... this can cause you problems in some situations when you want a “neutral” roll value in Pose Mode... in most cases you'll want to recalculate the roll value with Ctrl-N (which will remove the visible rotation and insert a value in the “Roll:” field in the Transform Properties subwindow) and give it a non zero roll value in Edit Mode so that in Pose Mode it is properly aligned and rotated for poses and constraints ...

You won't get that behavior if you rotate in either the top or side views if the bone is aligned to the view ...

It's just that if you have the Y axis (the axis the bone rolls on) perpendicular to the Global Y axis then you will have to recalculate the bone roll angles to get it aligned properly for pose space .

By “bone is aligned to the view” I mean what it is like when you first add a bone in a preset view, i.e. head at bottom and tip up at top in Edit Mode with no rotation applied .

<http://blenderartists.org/forum/showthread.php?t=110721>

A: Fligh If you look at your left hand, palm down, from above, and assume for this exercise that all four fingers have their X-Axis pointing towards the thumb and have a Roll Angle of zero (same as if you created an Armature in Top View), but the thumb would need a Roll Angle of about 90. So if you rotate the first digit of the fingers around their Local X (R-XX) they would go down perpendicular to the back of the hand, but if you rotate the thumb (Roll Angle 90 and again around the Local X) it would go Left, towards the fingers.

Ahh.... but who bothers to pose bones like that? You just grab the IK Target and move it!. Well the Transform Matrix is divided into four chunks (I think? I'm not sure of this but it helps to think of it [or look at it] this way and if you actually test it in Blender it works), Bone Space (the structure of the bone including Roll Angle), Armature Space (the structure of the Armature = the sum of all BoneSpace), Pose Space (transforms relative to Rest Position = recorded changes in BoneSpace within ArmatureSpace) and World Space (the sum of all the above relative to the (transforms of) the Armature Object).

<http://blenderartists.org/forum/showthread.php?t=102209>

A: Fligh Anyhow, In Blender, In Top View (always add armatures in Top view) the Z-Axis is Facing you and Y is Up. So when you add the first Bone it lies along the Y-Axis. You cannot change that parallel-to-Y and any future Bones you extrude will use that bones axis as reference. If you add an Armature in Front view the bone's (local) Z-axis will point to the Global -Y (Y-Negative)

and can lead to problems later on.

<http://blenderartists.org/forum/showthread.php?t=102609&highlight=bone+orientation>

A: Vertex Pusher Well I guess I may have a little something to contribute to this ... but I have been trying to learn the armature system in depth for the past few months and this particular issue came up with a complicated rig I was working with ... and I basically “discovered” that you really shouldn't have any other axis other than the Y as the “up” axis . And precisely because the bone roll angle issue .

It seems that despite how the view ports are configured (with the Z as the “up” axis) the default “up” axis is the Y ... for everything ... This is a bit annoying but if you have, like me, configured Blender to start with the front view open and have the Transform Properties window always open, you will see that everything when created in the front view is actually rotated 90 degrees along the X axis once you tab into Object Mode and that you need to Ctrl-A all objects so created (I guess Blender “out of the box” has the top view open ? ... it's been a while so I can't remember ...) . This isn't all that important for most things but for the armature system this does create a problem - bone roll values that you don't want .

Just as a simple test : Open up Blender and add an armature in the front view . Grab the tip and move it to a corner of the grid so that it is angled 45 degrees . Tab into Object mode . Open up the Transform Properties window . You will notice that your armature is rotated 90 degrees along the X axis . So ... Ctrl-A the armature to have it's “up” axis as Z ... And suddenly you will see the bone rotate along its roll axis . Tab back into Edit Mode, select the entire bone and Ctrl-N to recalculate the bone roll angles ... the bone will rotate back to the way it was ... but with a change in its roll value (-45 or 45 depending on which way it is angled) ... Now repeat the above but this time create the armature in top view ... And you will realize that despite being angle at 45 degrees to the view like above no roll value ... not even after recalculating the bone roll angles (Ctrl-N) ...

Now I am assuming that this occurs because the Y axis is the one used to calculate the bone's roll ... and that if you have anything other than the Y as the “up” axis the roll values get screwed up because now the roll axis (Y) is tangent to the up axis instead of being the up axis . Just turn on “Draw Axes” and look at what the “up” axis is for an individual bone is ... the Y .

And just another bit of proof ... The really great new “limit” constraints (loc,rot and scale) added in the last release ... Try and apply the limit rotation constraint to the first armature (the one with Z up and rotation applied) . Now turn LimitX on with no min or max values and do not turn on the “local” co-ordinate space ... it will fall flat along the Y axis as if created in the top view ... This is occurring because despite the fact that you have applied the 90 degree rotation along the X axis to the object, the

constraint is enforcing the global up axis ... which is ... the Y .

Now the only work around that I could come up with was to create an empty and parent a Y up armature to it then rotate the empty 90 degrees to be able to use the Z up view ports and also making sure the mesh also had the proper orientation as the armature . This prevents me from accidentally applying the rotation to the armature and I can use the numypad hotkeys to navigate the views Else you could just clear the roll values manually if you insist on having the Z as the up axis ... though you won't be able to get rid of the rotation that is caused by being tangent to the Y axis without odd roll values ...

Sorry for the long winded response, but this has been bugging me for a while ... Hopefully with the refactoring for v2.5 later this year there will be an option as to which axis will be the "up" axis with regards to the various views ..

<http://blenderartists.org/forum/showthread.php?t=93578>

5.61.11 General Armature Information

Armature Object is like any other object type: It has a center, a position, a rotation and a scale factor. It can be edited. It can be linked to other scenes, and the same armature data can be reused on multiple objects. All animation you do in object mode is only working on the object, not the armature's contents like bones. Link and Materials panel: The AR: field let you rename your armature Datablock. The dropdown is a quick way to select which Armature datablock you want to connect to this armature. You can keep more than one version for the same character. Useful when you have a special move to achieve in a shot, you can turn on an armature for a special purpose. The F button is an option to assign a Fake user to the Armature. Again if you have more than one armature for your character, it's a good idea to turn the Fake on, because if your armature datablock is not used (linked) it's not going to be saved in your .blend files. You can always do batch Fake-assignment of armatures by opening the Datablock browser (SHIFT-F4KEY), go in Armature datablock, select all the armatures you want to keep, and Press the FKEY. The OB: field is just to Rename your armature Object to something more cool and useful than Armature... Armature.001...

Delay Deform: This was useful before as the old system was very slow. What it does is when you do a manipulation to the rig, it waits until you finish to update the view. Can still be useful though.

Weight: This specifies how strongly this bone will influence the geometry around it, relative to the other bones. If two bones crossing each other, both with envelope influence, have the same weight (like 1:1) they will influence the surrounding geometry equally. But if you set one to 0.5, the geometry will be affected more significantly by

the other one, with weight 1. For example, in this image, 2 bones using envelope influence try to move the same geometry. The 2 on the left have the same weight, you can see the geometry didn't move. On the right, one of the bones has 0.5 so the bone with weight 1 is winning the tug-of-war!: Deform: This lets you say if you want the bone to deform the geometry at all. Switching it off is like setting the weight to 0, except it's faster this way. Useful when using a bone as a target or a controller, i.e. a bone you just want to use to control other bones, but not the geometry itself. Mult: to deform geometry you can use vertex group and/or Envelope. The ability to mix both of these methods is handy for using one to tweak the other. For example, you might use envelope everywhere but tweak difficult places manually with vertex group. We'll discuss this in more detail later on. There are two number fields to better tweak the effect of B-Bones. The in/out is used to tell the scale of the virtual handle of the bezier curve. In is the Root of the bone, and Out is the Tip. The bigger the value, the bigger the effect of rotation. You can do ALT-SKEY on one or more bones while in Envelope display mode to tweak the envelope size in real time while animating. Useful when for example you move the hand and some part of the character isn't in the influence zone; the result will be that some vertices will stay behind.

Tip: Bake envelope to vertex groups The workflow is very simple. When you are done with the envelope's tweaking and you have gotten the best out of it, delete the Armature modifier and parent the mesh to the armature(CTRL-PKEY). Parent it to "armature" when asked and "Create From Closest Bones". Do ALT-PKEY and redo the Armature modifier. Now all the envelope influence are converted to Vertex Groups. This way you can further tweak influence zone using Weight paint. More info in the following pages. You can edit this white zone in Editmode or posemode by going in Envelope display mode, selecting bones and using SKEY or ALT-SKEY The Mult option will tell Blender to multiply the weight it get from envelope (let say 0.7) with the weight you painted in weight paint (let say 0.5). The result will be $0.5 \times 0.7 = 0.35$ so in fact you just tweaked the envelope influence to 0.3 when it was at 0.7. If you don't want vertices to be part of the zone, you can always paint it with 0, as $0 \times (\text{something})$ will always give 0. This way you can give custom shape to your envelope. More on weight paint on next page.

Deform with only things weight painted Deform + Multiply with only things that are weight painted within the envelope radius

Process: Make bones. Make good envelope transformations(including some overlap). Alt+P to make virtual parent(from nearest bones) this will already weight paint parts based on envelope influence

"All faces" tells Blender if you want to paint on all faces in the mesh or just the visible one. "Vertex Dist" tell blender to use vertex distance instead of faces. When active, the

painting will only check if the vertex is in the brush, then apply a weight value. If it's off, all vertices part of the faces in the brush will receive weight value. Turning on Vertex Dist can give good results when you have a lot of polys in your mesh. Constraints are calculated from first to last. So if you have two Constraints working on the same channel, let say Location, The last one will most probably win the chance to move the object. But... Most of the constraints have influence slider to tell how much it influences on the stack. If the last constraint has an influence of 0.5 it will mix the result with the one before. Most of the time this little constraint is useful to stick objects to one another. By playing with the Influence you can tell when it will work, when it will remain motionless.

5.61.12 Building Armatures

Ludwig's Stretchy Spine In the spine Ludwig model keep in mind the following things. The SpineStretch.null is necessary. It must be set to Hinge and Deform to keep the bones from scaling. IKSpine must have Deform turned off(none selected). Spine 4 or the last bone child in the chain(closest to the neck) must have Deform turned off(none selected). Scaling problem is usually tied(maybe always) tied to an improper child/parent relationship to a bone that is involved with stretching.

Created a mirrored copy without using X-Axis Mirror select the bones that you want to mirror , put the 3D cursor at the center : And make shift + d , and directly , S , X , –1 ...

Other Information You can extrude a new bone from the selection using EKEY. This will create a bone connected to the original one, meaning the Root of the new bone will follow the Tip of the original one. You can also CTRL-LMB to extrude a new bone. It will extrude to where you clicked.

Alternatively, you can connect two existing bones by selecting them one after the other and pressing CTRL-PKEY. You can then choose either 'Connected' (the child bone - the one you selected second - will automatically be moved so that it touches the parent) or 'Keep offset'. aa Using the WKEY menu, You can subdivide your bone or flip the name of the bone between Left-Right (See Naming convention below). You can delete the bone with XKEY You can select a chain of bones (connected together) using LKEY, when you hover your mouse over a bone. In many cases, rigs are symmetrical and can be mirrored in half. In these cases, it is helpful to use a left-right naming convention. This is not only useful for your own sake, but it gives Blender a hint that there is a pair of equivalent bones, and enables the use of some very cool tools that will save you some significant work. It's helpful to name your bones with something useful telling you what it's there for, such as leg, arm, finger, back, foot, etc. If you get a bone that has a copy on the other side, however, like the arm (you have 2 arms right?), then the

convention is to call them arm.Left and arm.Right. Other alternatives are also possible, like _L, _LEFT, _left, .L, and .Left. Anyway, when you rig try to keep this left-right thing as accurate as possible; it will pay off later on. You can copy a bone named blah.L and flip it over using WKEY --> flip name. So the bone will be blah.L.001 after you copy it, and flipping the name will give you blah.R. Blender handily detects if the .001 version already exists, and increments the number for you.

5.61.13 Useful Rigs

Bassam's(slikdigit) Mancandy 1.0 <http://freefactory.org/posts/candy-for-everyone>

Bassam's(slikdigit) Mancandy 2.0 One of the most complex rig I have available for Blender. It has lattice based stretching. <http://freefactory.org/posts/candy-for-everyone>

Calvin's 05, 2017 This is an alternate foot roll method to Ludwig's rig.

Calvin's handy02

Clean3D's Mouserig <http://blenderartists.org/forum/showthread.php?t=79305>

Cognis's AnimTest Interesting use of empty's as a bone.

Daniel Martinez Lara's 3-plender leg This is a complex leg rig. It has many IPO driven bones. http://www.daniel3d.com/pepland/misc/3dstuff/blender/rig/3-plender_leg_v0-1.zip

Jason Pierce's(sketchy) Ludwig <http://jasonpierce.animadillo.com/resource/ludwig/ludwig.html>

Kugyelka's Walkcycle

Martin Georgiev's(Animarto) Walk_anim

Master Yoda

Michael Thoenes's Suzanne Rigged <http://creationanimation.com/sites/thoenes/animation/3dtutorials.htm>

Nathaniel Shaw's Generi

Noodlesgc's (Mike Roberts) W1r3z <http://www.esnips.com/doc/276d8bd1-0070-4ee4-bd8c-e787b507d6a9/W1r3z.blend>

Nozzy's Driven hand <http://blenderartists.org/forum/showthread.php?t=18754>

Nozzy's Skinny Guy <http://blenderartists.org/forum/showthread.php?t=30994>

Otothecleaner's Flor

Virgilio's Otto 1.6 Interesting constraint to keep the feet on the floor. http://uploader.polorix.net//files/99/otto_v1.6.zip

Rbackman's Bounce9

Rbackman's Zeyne

Simple's Character

Tugkan's Sharlo

Woodman5k's Bunny <http://web.pdx.edu/~{}wlf/Bunny.rar>

Yagapayanata's Rig

Yagapayanata's Rig(2, 4, 6 legs)

5.62 Customization

5.62.1 Changing the Theme

Like many other programs with a graphical user interface, Blender allows you to customize or modify many aspects of its appearance. A collection of such customizations is called a **theme**.

Theme controls are found in the User Preferences window. Click **LMB** on the “Themes” button to activate a drop-down menu which can be used to select different themes. Below this is an “Add” button that can be used to create new themes. If any themes have been added, a “Delete” button will appear, along with other controls to adjust the current theme.

This book presents screen shots using the default theme. If you are new to Blender, you should continue using the default theme as you progress through the book, to avoid confusion.

5.63 Mist - Make Objects Opaque

How Blender's Mist works If you render Objects within Blender's Mist, they mix up with the Background-Color, because Blender decreases their Alpha-Value by increasing Mist-Intensity and distance from the Camera.

The Problem This does a well job if your scene has a constantly-colored Background, but in case of a real Background or World-Texture it looks not convincing.

All objects are looking like ghosts.

The Solution explained briefly This page will show you two ways to get rid of that Problem. The major thought is to mix Objects with an opaque color, instead of making them transparent. The more Transparency, the more of the opaque color will be added.

Because Blender delivers a (dynamic) Alpha-Value with Mist enabled,

we can use this value as a factor for increase or decrease of mixing the objects with a certain color (or even to enhance or lower the contrast of materials, etc.)



Here's a render result with mist and the “Mixing-Technique”. You see, that you don't see shining the Background through.

The first solution goes with the help of an external compositing program and won't be explained in depth as we want to focus on how to solve this issue within blender since it is open source.

Solution 1: Make use of an external composite software.

Just render your Background and your Objects separately with Alpha-Channels.

Then you import these files into a composite program on two layers.

(One for the Objects and one for the Background). Between these layers you set a new “Color-Layer” with a solid greyish-blue color (because this fits the color of the atmosphere) and create a clipping mask.

A “clipping mask” means that the “Color-Layer” is only visible where the objects of the top layer are.

And where there's no object you can see the Background.

As the object is partially transparent (because of Blender's Mist),

you will see the greyish-blue color shining through.

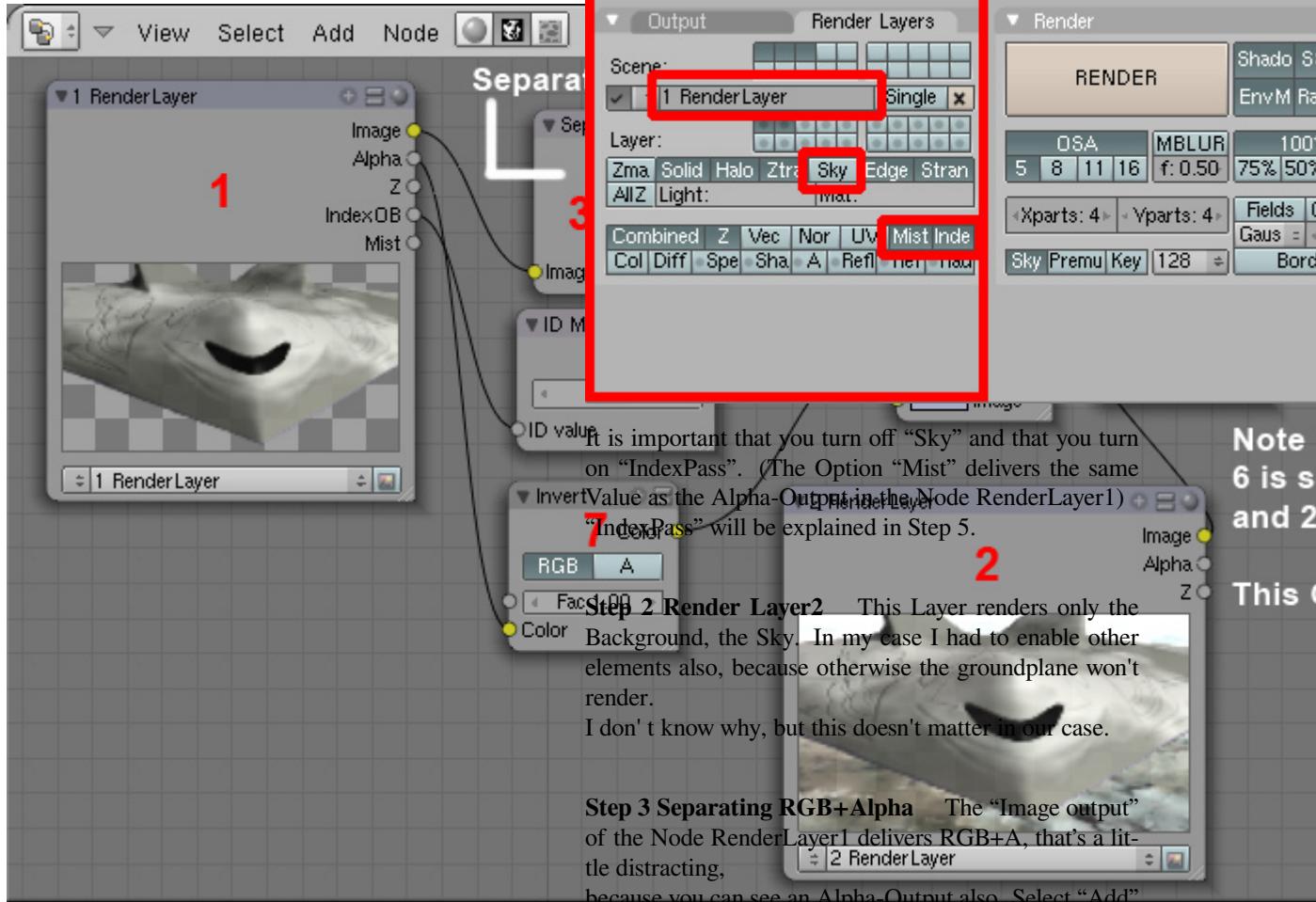
The more the object is away, the more transparent it is, and the more the color-Layer will shine through the Objects.

Solution 2: Make use of Blender's Composite-Nodes.

Since Solution 1 seems not to be a big deal as we cannot afford expensive packages, we are also able to achieve this “Mix-Technique” by Blender's built-in Composite-Nodes.

To continue reading it would be good, if you know a little about Blender's Compositing-Nodes.

Now, here's the interesting part of the tutorial, the Graph of the Nodes in the Node-Editor:



Description:

Step 1 Render Layer1 In the Render Settings Tab you assign the Objects which you don't want to be transparent to Render Layer1.

Step 2 Render Layer2 This Layer renders only the Background, the Sky. In my case I had to enable other elements also, because otherwise the groundplane won't render.
I don't know why, but this doesn't matter in our case.

Step 3 Separating RGB+Alpha The "Image output" of the Node RenderLayer1 delivers RGB+A, that's a little distracting, because you can see an Alpha-Output also. Select "Add" -> "Color" -> "Separate RGBA" in the Node-Editor Window.

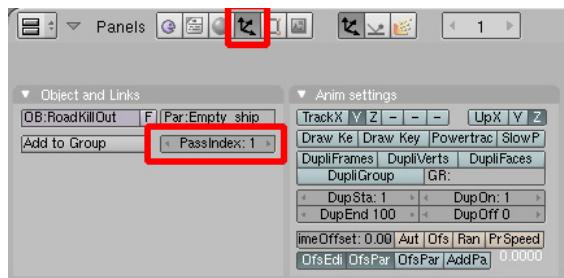
Step 4 Combining RGBA Now the Separated RGB gets combined again to RGB, but this time with a new Alpha of Step 5.

Step 5 Making the transparent Objects opaque
The good thing is, that Blender's Object-ID Alpha Value does ignore mist.
So we can add an ID-Mask Node to get the Alpha of the Object into the new Image.
Here you must set the ID of the Object to the same number as in the ID-Mask Node.
Select the Object and set its ID Pass-Index!
Beside is a screenshot where you set the Object ID.

Step 1 Render Layer1 In the Render Settings Tab you assign the Objects which you don't want to be transparent to Render Layer1.

See <http://wiki.blender.org/index.php/Doc:Manual/Render/Layers> for how to deal with RenderLayers.

Step 6 Mixing the Object with a Color Here you add a Mix-Node to mix the Object with a "Horizon-Color". The Color should be set to a bright greyish-blue as this is the color of the air inner the atmosphere.
Any color is possible, you may think of a dark red in a fire-hollow or of black if somewhere out in space.



In the red marked section of the buttons window, you can set the ID of the object

Step 7 Setting the Factor of how much of the color should be added Blender delivers an Alpha-Value to the Objects due to the Mist.

If an object is far away or if the Mist-Intensity is very high,

the Alpha-Value of the Object decreases, which means the Alpha-Value becomes more black.

And if the Object is close, it has accordingly high Alpha, which means getting white.

Black means a Value of R-G-B 0-0-0 and White a Value of R-G-B 1-1-1.

Since we want to use these values as a factor of how much to blend the Object with our "Horizon-Color", we need to invert this Value.

(Close Objects = High Alpha = Factor almost 1 - but we need Blend Factor close to 0, because we want low mist as the Object is close)

So let's add an "Invert-Color-Node".

Get the Settings of the Node-Graph for this Node.

This Node inverts Black to white, dark-grey to bright grey, etc.

We now use this inverted alpha-value (or Mist-value) as a factor for mixing the Horizon-Color with the Object.

Step 8 Layering the colored Objects over the Sky-Background Now you add an Alpha-Over-Node which will layer our Objects over the Sky-Background. Be aware to use the settings shown in the screenshot and that the order of the two Image-Inputs of the Alpha-Over-Node is very important. The second Input always gets over the first Input.

Step 9 Add an Output-Node Finally you need an Output-Node and to enable "Do Composite" in the RenderSettings.

Step 10 Example Here an Example, of how it worked out in my project:



- Close Distance - Almost no Mist



- Far Distance - Low Mist or atmospheric opacity



- Further Distance - (!)The Object is even scaled



- Farthermost Distance - In original size the Object would be so small that it isn't visible -only to make the Logic clear

Summary Be sure, you got every step right - then all will work out properly.

What is almost of more importance, that you understood the logic behind this, so that you learn about Blender's Compositing-Nodes and the 01-Logic ;)

Since "Mist" means something like "Bullsh**" in German I did not believe that it really is like that in Blender - and it isn't! Blender is a quite nice and powerful tool.
And I hope that it'll ever be Open-Source.

Chapter 6

Text and image sources, contributors, and licenses

6.1 Text

- **Blender 3D: Noob to Pro** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro?oldid=3286320 *Contributors:* Omegatron, DavidCary, Furrykef, Robert Horning, Alerante, Mats Halldin~enwikibooks, Geocachernemesis, Mkn, Argento, Panic2k4, Tualha, Qrc, Koos Jol, ManuelGR, StarryTG, Spiderworm, ZeroOne, Goeland86, Surphaze, Dan-aka-jack, Kellen, Jonon~enwikibooks, Uncle G, 01mf02, Serge, Wwwwolf, Bmud, Crouch, Popski, Yosofun~enwikibooks, Saysaknow, Astronouth7303, Derbeth, Jclee, Snarius~enwikibooks, DuLithgow, Wikibooks is Communism, Trident~enwikibooks, Camel, Gabriel~enwikibooks, Ben Russell, EatMyShortz~enwikibooks, Allefant~enwikibooks, Darklama, Oracleoftruth~enwikibooks, Jmartens, RobinH, 1983, Randolph Richardson~enwikibooks, Rocastelo, AM088, Dragontamer, Soylentgreen, Leinad13, Whiteknight, Alabandit, Chw333, Ideasman42, Pld~enwikibooks, Crawmm, Wavez, Klowner, Joe 042293, WeirdHat, Kernigh, ParallaxTZ, Darkonc, Xerol, Jomegat, Iluvblender, Greensweater, Osxrules, SchmS, Gabio, David Osborn, Vault, Super3boy, Milky~enwikibooks, Mastermind 007, Halley, Pietrodn, BimBot, Jguk, Joystick Game, Hagindaz, Joystick Tutorial, Wcsc, DanMan, Fr~enwikibooks, Ordigdug, Orbisonitrum, Zaknafien~enwikibooks, Jogai, Jawboot, Mdd4696, LION~enwikibooks, Sumoncil, Webaware, AndyD, Aragan Jarosalam~enwikibooks, Cleobod, Wynmmc, Aidan.Sullivan, Cürps, Scottpledger, Damyeon, Freakazo v2.1, Arch dude (usurped), Justthisguy, Lee Carre, Dirk Gently~enwikibooks, Charybdis~enwikibooks, Swift, Bootaleg, Skyguy, Az1568, Ikarsik, Nivix, Tannersf, Xania, Read-write-services, Herbythyme, Rm5248, NCS Digiboy, Romainbehar, DanAvner, Moreejt, Twentytortures, Hrakaroo, Zeviion, Dolsson5, AerieZen, AdRiley, Rws~enwikibooks, BeHE, Gogreen, Lazy-lump, Jamin3D~enwikibooks, Venar303, Dan13, TBOL3, Boyage, Notmyopinion~enwikibooks, Eb264, Quantum Anomaly, Tedka, Mike.lifeguard, MSK61, LokiClock, Damian Yerrick, Remi, EvanCarroll, Michaelx153, Recent Runes, Mike's bot account, SergeantOreo, Adolfohm, Neo5842, Moohasha, Grapw the Giant~enwikibooks, Sailer, Chelseafan528, IJstLrndAlotAbout-StargateAtWP, Morerunes, Pi zero, TwistedWeasel, Ramac, Neoptolemus, Malaz, Robert Valík, Thecheet123, Noob1230, Big Bill57, Mach1723, Embri, Soeb, Jackechan, ThomasShepard, Xemkis12, Radialronnie, Geekboy2000, Goodgirl4, Dokuro, LightningIsMyName, OekelWm, Rivelozo, Waldir, SoylentGreen, CarsracBot, Dallas1278, Turtle Man, BobbyMatt, Miyazaki~enwikibooks, Kayau, Bullercruz1, Mrkat, Tom soderlund~enwikibooks, Sureshgn, Magicink, Cbh, Nuclearpidgeon, James.Denholm, Billinghurst, QuiteUnusual, Nick.anderegg, AttilaMH, Tempodivalse, BANZ111, Sigma 7, Adrignola, Arch dude, Zilch~enwikibooks, Null Point, Rosver~enwikibooks, Sendoshin, Electro, GeForce3, Tyrant monkey, Ftiercel, Camp, Whitepaw, Jona, Harry107, Morrolan, Tjb0607, Dobz, Woodsocket, Rwxrwxrwx, Iodrome, Admirj, Wentam, 492star, Jywhtj, Dirk Hünniger, Reyk, Admiral Spencer, Mortense, Hoo man, Pearts, Ldo, Bourriquet 42, JimRester, Stepheng3, Stryn, Frigoton, Vi3gamehkr, Invoker~enwikibooks, Othmanskn2, JamesNZ, Guoyunhebrave, XDraconian, ~riley, VB4231, Cubicsphere, Vesseshhebbar, Buttflyhoney, MercedMike, Syum90, Vvpinks, Animajosser, Kidkkr, M.K. Tee, Booked Dude and Anonymous: 504
- **Blender 3D: Noob to Pro/Unit 1: Knowing Before Making** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Unit_1%3A_Knowing_Before_Making?oldid=3227957 *Contributors:* Nanobug, Panic2k4, Jomegat, Lee Carre, Damian Yerrick, Bullercruz1, Sigma 7, M.Nießen, Electro, Metahop, John.delannoy, SmileyCactus, Rwxrwxrwx, Fishpi, Ldo, JimRester, India103, Stepheng3, Booner~enwikibooks, Hash~enwikibooks, Nkansahrexford, VB4231, Abramsky, Joel Sjögren, ErrydayLearning, Spekular, Pjhanzlik, Kidkkr and Anonymous: 24
- **Blender 3D: Noob to Pro/What Blender Can Do** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/What_Blender_Can_Do?oldid=3130978 *Contributors:* Lee Carre, Ldo, Stepheng3, Downdate, Syum90, Spekular and Anonymous: 10
- **Blender 3D: Noob to Pro/3D Geometry** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/3D_Geometry?oldid=3184665 *Contributors:* Xania, Jeff G., Bullercruz1, Tsester, Adrignola, Electro, Rwxrwxrwx, Ldo, Stepheng3, Mtjlt, Marc-André Aßbrock, AmieKim, 4thought, Nkansahrexford, AllenZh, JustinTime55, Jay.editor, Smjefferson602 and Anonymous: 31
- **Blender 3D: Noob to Pro/Coordinate Transformations** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Coordinate_Transformations?oldid=3164127 *Contributors:* Ldo, Greenbreen, Kerina yin, Nkansahrexford, Jay.editor, Learnerktm and Anonymous: 11
- **Blender 3D: Noob to Pro/Orthographic Views** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Orthographic_Views?oldid=3201908 *Contributors:* BANZ111, Ghiraddje, Electro, Avicennasis, Ldo, Stepheng3, Greenbreen and Anonymous: 3
- **Blender 3D: Noob to Pro/Perspective Views** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Perspective_Views?oldid=3166398 *Contributors:* SoylentGreen, BANZ111, Zilch~enwikibooks, Electro, Ldo, Stepheng3, Greenbreen and Anonymous: 6

- **Blender 3D: Noob to Pro/Coordinate Spaces in Blender** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Coordinate_Spaces_in_Blender?oldid=3166407 *Contributors:* Lee Carre, SoylentGreen, Bullercruz1, Adrignola, Arch dude, Tigerstick-figure, Squaremo, Electro, MC10, Radly~enwikibooks, Ldo, Patrolboat, Stepheng3, Greenbreen, Jay.editor and Anonymous: 22
- **Blender 3D: Noob to Pro/User Interface Overview** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/User_Interface_Overview?oldid=3306598 *Contributors:* Bullercruz1, QuiteUnusual, Sigma 7, Manequinho, Electro, JHeathe, RoseCityRemona, Stepheng3, Greenbreen, JamesNZ, Nkansahrexford, Rgrasmus, Breroner, Katastrov, Pjhanzlik, Hugoxxx and Anonymous: 19
- **Blender 3D: Noob to Pro/Keystroke, Button, and Menu Notation** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Keystroke%2C_Button%2C_and_Menu_Notation?oldid=3166425 *Contributors:* Qrc, Spiderworm, Crouch, Popski, Derbeth, Jclee, Rylz~enwikibooks, Jessemerriam~enwikibooks, Loplin~enwikibooks, Jguk, Rabenschwinge~enwikibooks, Ordigdug, DavidJaquay, Dennis G. Jerz~enwikibooks, N comme Nul, Justin6300~enwikibooks, Casimir~enwikibooks, Romainbehar, Acsteitz~enwikibooks, Dolsson5, Slady, Damian Yerrick, EvanCarroll, Mecalith~enwikibooks, Polluks, Tyaedalis, ThomasShepard, Bullercruz1, Adrignola, Clemens~enwikibooks, Habstinat, Ldo, Stepheng3, Hom sepantha~enwikibooks, Harrybrowne1986, Greenbreen, JamesNZ, Nkansahrexford, Wiz3kid and Anonymous: 65
- **Blender 3D: Noob to Pro/Non-standard equipment** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Non-standard_equipment?oldid=3166431 *Contributors:* Mxn, Jclee, Jomegat, Lee Carre, EvanCarroll, Tyaedalis, Spartacus3d, Θεόφιλε~enwikibooks, Adrignola, Velociostrich, Stepheng3, Qwertyyzz18, MasterSkrat and Anonymous: 28
- **Blender 3D: Noob to Pro/Operating System specific notes** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Operating_System_specific_notes?oldid=3211837 *Contributors:* Lee Carre, Rory O'Kane, EvanCarroll, Dlr, Tyaedalis, Soeb, Brian 252, Spartacus3d, Adrignola, Sendoshin, Duplode, Selkovjr, Ldo, Stepheng3, Marc-André Aßbrock, Hbrown101, Quantumavik, XDraconian, Danibits, Hmsousa and Anonymous: 20
- **Blender 3D: Noob to Pro/Blender Interface** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Blender_Interface?oldid=3179549 *Contributors:* Qrc, Spiderworm, ZeroOne, Zoohouse, Crouch, Popski, Tedesson, Hagindaz, N comme Nul, Soeb, Adrignola, Ldo, Stepheng3, JamesNZ, AtheneNoctua, Erpants, Spekular, Spamm and Anonymous: 24
- **Blender 3D: Noob to Pro/Blender Windowing System** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Blender_Windowing_System?oldid=3169240 *Contributors:* Furykef, Geocachernemesis, Argento, Tualha, Qrc, Spiderworm, ZeroOne, Crouch, Popski, Derbeth, Ruyven~enwikibooks, Lynx7725, Oracleoftruth~enwikibooks, Goncalopp, AM088, MechaBlue, Ideasman42, Sterre~enwikibooks, Jomegat, Jguk, Gavinmusic~enwikibooks, Mdd4696, DavidJaquay, Thesandlord~enwikibooks, Gijs.peek, Az1568, Xania, Rm5248, DariusWiles~enwikibooks, Edmetric, UsedHONDA, Winterbourne, Recent Runes, Geigerho, Pedro Fonini, Filaminae, Lambda (Usurped), Soeb, Jamessnell~enwikibooks, Pruit K., Xlinuxkdex, El imp, MacPhyle, Adrignola, Null Point, Ivucica, Diblidabiduu, Rjh009, Andrewtrefethen, Ldo, Stepheng3, Douglasfb15, Pignol23, Kerina yin, Rltromble, Inquisitor Kaoz, JamesNZ, Red-confetti, Nkansahrexford, Spacanut42, Frullatore, Visually, TROPtastic, Spekular, Anoadaragon453, Kjartan93 and Anonymous: 201
- **Blender 3D: Noob to Pro/User Preferences Windows** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/User_Preferences_Windows?oldid=3166837 *Contributors:* Spiderworm, Jomegat, ShakespeareFan00, Lazy-lump, Yhevhe, Bekenn, EN-Eagle 101, Blenderizer, Ramac, Soeb, 5o~enwikibooks, Hxcan, Shegelmie, Aneesml, Ldo, Stepheng3, Inquisitor Kaoz, JamesNZ, Visually, TROPtastic and Anonymous: 24
- **Blender 3D: Noob to Pro/Properties Window** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Properties_Window?oldid=3283133 *Contributors:* DavidCary, Geocachernemesis, Panic2k4, Tualha, Qrc, Spiderworm, ZeroOne, JiBril, Crouch, Popski, Smarius~enwikibooks, Withinfocus, MechaBlue, Tedesson, Webaware, Arch dude (usurped), TBOL3, Yes Man~enwikibooks, Mouse among men~enwikibooks, Imaginationac, Savage Beef, ThomasShepard, QuiteUnusual, Tsester, Θεόφιλε~enwikibooks, Adrignola, Shegelmie, Ldo, Stepheng3, Greenbreen, JamesNZ, Redyoshi49q, Kbellis, Ben Colley and Anonymous: 74
- **Blender 3D: Noob to Pro/3D View Windows** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/3D_View_Windows?oldid=3208210 *Contributors:* Qrc, Spiderworm, Jonon~enwikibooks, Odie5533, Crouch, Popski, Cale~enwikibooks, Boris-blue, David Cochran, Darklama, AM088, Jomegat, Michael Coupland~enwikibooks, Marty one, Banzai~enwikibooks, Jollyroger, Xania, Herbythyme, Rm5248, Acsteitz~enwikibooks, DariusWiles~enwikibooks, Lazy-lump, Bekenn, Darkfire, Sevesik, FriedChiliDog1, Garmon Sutrix, YesPoX~enwikibooks, Itismike, EvanCarroll, Dagnode~enwikibooks, EN-Eagle 101, DandellionKimbam, ThomasTenCate, Soeb, Jeremy, Jackechan, ThomasShepard, WesleyJL, 2DCube, Atarun, Sigma 7, Gairlochan~enwikibooks, Adrignola, Arch dude, Ivucica, Dougc, Wgoelkel, Mikespedia, Sestina~enwikibooks, Zbrown, Entera voss, Albjimrui, Andrewtrefethen, Dual Shock 2 ps2 controller, Ldo, Stepheng3, Robbots, SumOfwHoly, Nicoguardo, JamesNZ, Matt Pellegrini, BlenderDocs, Maniandram01, William Di Luigi, Rgrasmus, Wiz3kid, Visually, Arenguezus, Kjartan93 and Anonymous: 283
- **Blender 3D: Noob to Pro/Object Mode** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Object_Mode?oldid=3298329 *Contributors:* JackPotte, Tsester, Ldo, Craigallan.za, Quantumavik, Cubicsphere, TROPtastic, Fred dot u, SalathielGenese and Anonymous: 18
- **Blender 3D: Noob to Pro/Mesh Edit Mode** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Mesh_Edit_Mode?oldid=3199377 *Contributors:* Shegelmie, Mortense, Ldo, WataruA1, Vinnyviking, Kbellis, Cubicsphere, Wmjbean, TROPtastic, LysolPi-onex, Federhalter, Syum90 and Anonymous: 20
- **Blender 3D: Noob to Pro/Normals and Shading** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Normals_and_Shading?oldid=3220056 *Contributors:* Ldo and Anonymous: 7
- **Blender 3D: Noob to Pro/More Mesh Editing Techniques** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/More_Mesh_Editing_Techniques?oldid=3168675 *Contributors:* Ldo, Zephyrus Tavvier and Anonymous: 5
- **Blender 3D: Noob to Pro/Quickie Lighting** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Quickie_Lighting?oldid=3284770 *Contributors:* Ldo, Rgrasmus, Jason Olshefsky and Anonymous: 3
- **Blender 3D: Noob to Pro/Quickie Model** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Quickie_Model?oldid=3173122 *Contributors:* Geocachernemesis, Argento, Qrc, Spiderworm, ZeroOne, Zoohouse, Crouch, Popski, Cale~enwikibooks, Oracleoftruth~enwikibooks, Beetlemaniac~enwikibooks, Ebswift~enwikibooks, Jomegat, Mdd4696, Siggimund, Lee Carre, Rdp, Romainbehar, Slady, Darkfire, Obiesel, ThomasTenCate, Soeb, JoelGodin, Marco Guardigli~enwikibooks, Adrignola, Zilch~enwikibooks, Ivucica, Squaremo, Mikespedia, Rxwxrxwx, Andrewtrefethen, Ldo, Flavoice~enwikibooks, Stepheng3, Marc-André Aßbrock, JamesNZ, AmieKim, YXie and Anonymous: 89

- **Blender 3D: Noob to Pro/Quickie Render** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Quickie_Render?oldid=3173104 *Contributors:* Argento, Tualha, Qrc, Spiderworm, ZeroOne, Popski, Astronouth7303, Gumba gumba, Cale~enwikibooks, Rylz~enwikibooks, Michaelnelson~enwikibooks, Ebswift~enwikibooks, Jomegat, Thzi~enwikibooks, Super Dave~enwikibooks, Lee Carre, Romainbehar, Nowic, Darkfire, Blondandy~enwikibooks, SergeantOreo, ThomasTenCate, Soeb, Coffeeparis, JoelGodin, Spongebob88, Adrignola, Null Point, Irakrakow, TimDumol, Rwxrwxrxw, Ldo, Stepheng3, JamesNZ, AmieKim, Resul4e, Bobj~enwikibooks, XDraconian, Davekeiser, Vinnyviking and Anonymous: 69
- **Blender 3D: Noob to Pro/World Settings** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/World_Settings?oldid=3173056 *Contributors:* Ldo and Anonymous: 2
- **Blender 3D: Noob to Pro/Understanding the Camera** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Understanding_the_Camera?oldid=3173064 *Contributors:* Ldo, MacPepe, Nkansahrexford, AtheneNoctua, Serpinium, Gqfreeman, Ulzha, Tsharky87 and Anonymous: 7
- **Blender 3D: Noob to Pro/Improving Your House** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Improving_Your_House?oldid=3173068 *Contributors:* Panic2k4, Xania, Turtle Man, QuiteUnusual, Arch dude, Reyk, Ldo, Stepheng3, Douglasfeb15, Yhipavahit, SK123r, JamesNZ, AmieKim, Bamacs guy, Falstaff~enwikibooks, Oshaylinux, XDraconian, Kekie1, AtheneNoctua, Seejork, William Di Luigi, TROPtastic, Serpinium, Orphanlast and Anonymous: 17
- **Blender 3D: Noob to Pro/Modeling a Simple Person** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Modeling_a_Simple_Person?oldid=3174514 *Contributors:* Mats Halldin~enwikibooks, Geocachernemesis, Bijee~enwikibooks, Panic2k4, Spiderworm, ZeroOne, Pwd~enwikibooks, Crouch, Popski, Yosofun~enwikibooks, Cale~enwikibooks, DuLithgow, Borisblue, Rylz~enwikibooks, Allefant~enwikibooks, Schultz.Ryan~enwikibooks, 1983, Randolph Richardson~enwikibooks, AM088, Ebswift~enwikibooks, Kernigh, Jomegat, Citizen428~enwikibooks, Shinmoos~enwikibooks, Jguk, Nessup~enwikibooks, CaptBobo, Az1568, Rdp, Jyril, Xania, Herbythyme, Rm5248, Romainbehar, Mosquito~enwikibooks, Lebigot, AdRiley, CharlesLambert, Reece, Slady, Jeromerobert~enwikibooks, Pquijal, Tayewiki, IamInnocent, Darkfire, Mike.lifeguard, YesPoX~enwikibooks, Gilgamesh007~enwikibooks, Pedro Fonini, Wikidieter, Atimmer, Rreutel, JoelGodin, James.Denholm, QuiteUnusual, Argav, Sigma 7, Adrignola, Null Point, Ivucica, Robsterbertling, Squaremo, Wimijongman, Bonapon, HWV258, TimDumol, Ditw0101, Reyk, Shinydarkrai94, Pearts, Ldo, Stepheng3, Mtjlt, Rejiquar, JamesNZ, AmieKim, Corn Chowder, Jake.lewis4, Serpentine Cougar, William Di Luigi, Gqfreeman and Anonymous: 234
- **Blender 3D: Noob to Pro/Detailing Your Simple Person 1** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Detailing_Your_Simple_Person_1?oldid=3196983 *Contributors:* Maveric149, Argento, Sjc~enwikibooks, Spiderworm, Crouch, Yosofun~enwikibooks, Astronouth7303, Gumba gumba, Bacon~enwikibooks, DuLithgow, AM088, Withinfocus, Ebswift~enwikibooks, Jguk, Lapinbleu, DavidJaquay, FishFace, Dirk gently~enwikibooks, Bronzed bison, Hippopotamuses, Az1568, Rm5248, Lebigot, Mike.lifeguard, Phaedriel the Vandal, Mr Me, Someguywithnothingtodo, Atimmer, James.Denholm, Adrignola, Arch dude, Shinydarkrai94, Ldo, Stepheng3, Mtjlt, JamesNZ, AmieKim, Corn Chowder, XDraconian, Jake.lewis4, Gqfreeman, Schvenk, Phen17 and Anonymous: 111
- **Blender 3D: Noob to Pro/Detailing Your Simple Person 2** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Detailing_Your_Simple_Person_2?oldid=3185911 *Contributors:* Geocachernemesis, Argento, Panic2k4, Spiderworm, ZeroOne, Crouch, Popski, Yosofun~enwikibooks, Derbeth, Geri~enwikibooks, Bturnip, Joda~enwikibooks, Simon W, Pikpus, EatMyShortz~enwikibooks, Winters~enwikibooks, Mercator~enwikibooks, Oracleoftruth~enwikibooks, Thunder~enwikibooks, AM088, Spiffyandy, Ebswift~enwikibooks, Jomegat, Dessydes~enwikibooks, Jguk, Reilithion, Matt Howard, Mdd4696, Enselic~enwikibooks, Webaware, Bgks, N comme Nul, Mikegr, Hippopotamuses, Tannersf, Pinkmouse~enwikibooks, Rm5248, Romainbehar, Twofingers, Reece, Pootworm, SixFt12, Mike.lifeguard, Gilgamesh007~enwikibooks, Satrife101, Ramac, Happy2pester, CodeSnorter, Rheinhardt2, Simeon, Jtonti, QuiteUnusual, Nick.anderegg, Adrignola, Fede.Campana, Zilch~enwikibooks, Null Point, ReKast, Bonapon, Cosworth~enwikibooks, Yesuto15, Pearts, Ldo, Stepheng3, Mtjlt, JamesNZ, AmieKim, Corn Chowder, Memiles, PorsHammer, Cubicsphere, TROPtastic, Gqfreeman, Schvenk and Anonymous: 196
- **Blender 3D: Noob to Pro/Creating a Simple Hat** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Creating_a_Simple_Hat?oldid=3075647 *Contributors:* Esben~enwikibooks, Argento, Spiderworm, ZeroOne, Wwwwolf, Crouch, Popski, TiM-och, Yosofun~enwikibooks, Derbeth, Bturnip, Oracleoftruth~enwikibooks, Thunder~enwikibooks, Sledz41, Michaelnelson~enwikibooks, Pinzo~enwikibooks, Ebswift~enwikibooks, Miga~enwikibooks, Jguk, Hagindaz, Quantufinity@gmail.com, Digitalartist~enwikibooks, Mimooh, Swift, N comme Nul, Xania, Romainbehar, 3DSki, Mike.lifeguard, YesPoX~enwikibooks, Javac~enwikibooks, Gilgamesh007~enwikibooks, Jackechan, CodeSnorter, LethalReflex~enwikibooks, Bruce89, Adrignola, Fede.Campana, Irakrakow, Mr Speaker, Lionelb, Yesuto15, Hoo man, Pearts, Ldo, Bk314159, Stepheng3, Mtjlt, JamesNZ, AmieKim, Falstaff~enwikibooks, Kekie1, Davekeiser, Niflheim~enwikibooks, Cubicsphere, TROPtastic, Serpinium, Gqfreeman, Ulzha, Xivi-xxii and Anonymous: 145
- **Blender 3D: Noob to Pro/Putting Hat on Person** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Putting_Hat_on_Person?oldid=3186415 *Contributors:* Spiderworm, Popski, Socratesone, Yosofun~enwikibooks, Dalhamir~enwikibooks, Astronouth7303, Winters~enwikibooks, Oracleoftruth~enwikibooks, Withinfocus, Miga~enwikibooks, Jguk, SatanClaus~enwikibooks, Mdd4696, Phonuz, N comme Nul, Twofingers, Girdi, Bullercruz1, Rpyle731, Nick.anderegg, Adrignola, Zilch~enwikibooks, Entera voss, Avicennasis, Pearts, Ldo, Stepheng3, Mtjlt, JamesNZ, AmieKim, Burpen, Cubicsphere, TROPtastic, Serpinium, Gqfreeman, Mtby and Anonymous: 74
- **Blender 3D: Noob to Pro/Materials and Textures** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Materials_and_Textures?oldid=3013041 *Contributors:* Geocachernemesis, Spiderworm, Crouch, Popski, SchmS, Accurrent~enwikibooks, Xania, SoylentGreen, Bullercruz1, Adrignola, Merrilee, Reyk, Ldo, Stepheng3, Nkansahrexford, VB4231, Gqfreeman, Xivi-xxii and Anonymous: 15
- **Blender 3D: Noob to Pro/Quickie Material** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Quickie_Material?oldid=3263029 *Contributors:* Qrc, Spiderworm, Jonon~enwikibooks, Uncle G, Crouch, Popski, Wikibooks is Communism, Gregggreg, Knight13, SchmS, Pete xp, Mdd4696, Reverzentalive, Az1568, Siesta~enwikibooks, Xania, Beuc, Brinman, Malvineous, Edr~enwikibooks, NullPtr, SoylentGreen, Fezz~enwikibooks, Sinistra D32, QuiteUnusual, Spellmaker, Adrignola, Arch dude, Merrilee, Ldo, Stepheng3, Marc-André Aßbrock, JamesNZ, Bocercus, Resul4e, Falstaff~enwikibooks, FranklyMyDear..., Drblitzkrieg, Nkansahrexford, Cubicsphere, Syum90, Gqfreeman, Tghrtngqornegoqpuirnq, PokestarFanBot and Anonymous: 49
- **Blender 3D: Noob to Pro/Multiple Materials Per Object** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Multiple_Materials_Per_Object?oldid=3306415 *Contributors:* Kwi, Avicennasis, Ldo, Stepheng3, Freis~enwikibooks, JamesNZ,

Yuyuyak, Falstaff~enwikibooks, Nkansahrexford, Casper da Costa-Luis, Burpen, Cesarespcab~enwikibooks, Greenlid, Nobody231, Serpinium and Anonymous: 10

- **Blender 3D: Noob to Pro/Metal Versus Plastic** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Metal_Versus_Plastic?oldid=2553200 *Contributors:* Ldo
- **Blender 3D: Noob to Pro/Texture Settings** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Texture_Settings?oldid=3136157 *Contributors:* Ldo, MasterSkrat and Anonymous: 1
- **Blender 3D: Noob to Pro/Image Textures** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Image_Textures?oldid=2768107 *Contributors:* Nanobug, Spiderworm, SchmS, Rm5248, MSK61, SoylentGreen, Bullercruz1, Adrignola, Tjb0607, Ldo, JamesNZ, Davekeiser, Anonymous man, TROPtastic, Leaderboard and Anonymous: 11
- **Blender 3D: Noob to Pro/Procedural Textures** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Procedural_Textures?oldid=3202305 *Contributors:* Nanobug, Spiderworm, Derbeth, Soriyath, Greggreg, AM088, SchmS, Jguk, Mfox-dogg~enwikibooks, Bullercruz1, Nuclearpidgeon, Adrignola, Rwxrwxrwx, Ldo, Mhoram, Falstaff~enwikibooks, Davekeiser, Niflheim~enwikibooks, TROPtastic, Serpinium, Gqfreeman and Anonymous: 20
- **Blender 3D: Noob to Pro/Quickie Texture** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Quickie_Texture?oldid=3185895 *Contributors:* Spiderworm, Jonon~enwikibooks, 01mf02, Crouch, Popski, Allefant~enwikibooks, AM088, SchmS, Siesta~enwikibooks, Beuc, Plasmasphere, Brinman, Malvineous, NullPtr, SoylentGreen, Nuclearpidgeon, Xlinuxkdex, Adrignola, John Nagle, Bonapon, Rwxrwxrwx, Kenaku, Ldo, Weeix, Behemoth14, Pookz, TROPtastic, Nabeelfarooqui, Renagon Poi and Anonymous: 25
- **Blender 3D: Noob to Pro/Halo Materials** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Halo_Materials?oldid=3109355 *Contributors:* Beuc, SoylentGreen, Nuclearpidgeon, Adrignola, Ldo, Stepheng3, Pookz, Amigos, Animajosser, Xivi-xxii and Anonymous: 12
- **Blender 3D: Noob to Pro/Blender Memory Management** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Blender_Memory_Management?oldid=3177129 *Contributors:* The enemies of god, Rwxrwxrwx, Avicennasis, Ldo, Falstaff~enwikibooks and Anonymous: 5
- **Blender 3D: Noob to Pro/Bones** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Bones?oldid=3306347 *Contributors:* Derbeth, Oracleoftruth~enwikibooks, Jmartens, SchmS, Jguk, Rm5248, Niks1024, SixFt12, ThomasB~enwikibooks, AndrewBuck, Wim b, Root~enwikibooks, Bullercruz1, QuiteUnusual, Argav, Lordelph~enwikibooks, Adrignola, Arch dude, Mortennobel, Jake Scordato, Dogwynn, Rwxrwxrwx, Avicennasis, Mikenube, Pearts, Ldo, Stepheng3, Marc-André Aßbrock, Pookz, Mhoram, Serpentine Cougar, Jason Olshefsky, TROPtastic, Ulzha, Animajosser, Xivi-xxii and Anonymous: 65
- **Blender 3D: Noob to Pro/Mountains Out Of Molehills** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Mountains_Out_Of_Molehills?oldid=3177991 *Contributors:* Iamunknow, Spiderworm, ZeroOne, Derbeth, Camel, Megaloman~enwikibooks, Tpower~enwikibooks, AM088, Valid User, Beetlemaniac~enwikibooks, Jomegat, Neko18~enwikibooks, Mdd4696, Radiat-r~enwikibooks, N comme Nul, Tannersf, Moreejt, Twofingers, Lirtferk, Darkfire, Skeep, Manx61, Pedro Fonini, Obiesel, Newbienmiss, Joeblends, Ascen, ThomasTenCate, Jackechan, Blueskies~enwikibooks, Bullercruz1, Rajeshja, Spongebob88, QuiteUnusual, Argav, Adrignola, Fede.Campana, Kindeller, Zilch~enwikibooks, N00b Henry, Khono, Bonapon, Daworm, Hendric~enwikibooks, Rwxrwxrwx, Avicennasis, Yesuto15, Pearts, Ldo, Stepheng3, DbD, RVRegeek, Pookz, Slayerjp, Koltonaugust, MS10EL, TROPtastic, MasterSkrat, LesleyLai, Serpinium, Gqfreeman, Brimbard, Animajosser and Anonymous: 152
- **Blender 3D: Noob to Pro/Modeling a volcano** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Modeling_a_volcano?oldid=3197500 *Contributors:* Spiffyandy, Tayewiki, Scar~enwikibooks, Hobo Joe~enwikibooks, Adrignola, Illusionist~enwikibooks, Valdemarasl, Morrolan, Bonapon, Fedexdoom, Jfireball66, Rwxrwxrwx, Avicennasis, Falke~enwikibooks, Yesuto15, Reyk, Shinydarkrai94, Pearts, Ldo, Gaandolf, Pookz, LXXXV, Mikerancourt, SgtBursk, Bcroner, Gqfreeman, Animajosser, Mtby, Phen17 and Anonymous: 54
- **Blender 3D: Noob to Pro/Penguins from spheres** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Penguins_from_spheres?oldid=3197508 *Contributors:* Esben~enwikibooks, Argento, Panic2k4, Jomegat, Jguk, Domthedude001, Orbisonitrum, Jogi, Jawboot, Funky weasel, Norleaf, Cleobod, Adamamp~enwikibooks, N comme Nul, Herbythyme, Romainbehar, ShakespeareFan00, Twofingers, Reece, ProfoX~enwikibooks, CommonsDelinker, TBOL3, Shythinker, Boyage, Josellis, Mike.lifeguard, Gilgamesh007~enwikibooks, Mecolith~enwikibooks, Opa God, Eruantalon, Rekov, Ascen, Root~enwikibooks, CodeSnorter, Beginner~enwikibooks, Spongebob88, Rxy, DRNZ, Lokimaros, Adrignola, Brandished, Donlemay, Mattiasl, Camp, Bonapon, Physicsdude, Fidoda~enwikibooks, Mast3rlinx, Reyk, NinjaSaru, Ozgrin, Pearts, Ldo, Ibid49, Gaandolf, Recall~enwikibooks, Koltonaugust, SgtBursk, Gqfreeman, Ah3kal, Animajosser, Phen17 and Anonymous: 148
- **Blender 3D: Noob to Pro/Dicing With Depth** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Dicing_With_Depth?oldid=3178822 *Contributors:* Panic2k4, Chazz, QuiteUnusual, Rwxrwxrwx, Ldo, Lcos7, Tom990, Animajosser, Tazling and Anonymous: 14
- **Blender 3D: Noob to Pro/Model a Goblet** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Model_a_Goblet?oldid=3039787 *Contributors:* Adrignola, Arch dude, Avicennasis, Ldo, SgtBursk, Gqfreeman, Animajosser and Anonymous: 2
- **Blender 3D: Noob to Pro/Model a Silver Goblet** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Model_a_Silver_Goblet?oldid=3181350 *Contributors:* Clone Dad, Helderpc~enwikibooks, Jguk, Nanodeath~enwikibooks, Sandothergrade~enwikibooks, Dooglus, Wynnmee, Lsmoura, Az1568, AerieZen, Laleena, MSK61, Chuey, Brylie, Eyeonus, Pedro Fonini, Ramac, Prydain55, Abcdaa, Ascen, ThomasTenCate, Turtle Man, Argav, Yoshi1476, Adrignola, Sesu Prime, RavenWhitehawk, Camp, Rwxrwxrwx, Avicennasis, Yesuto15, Reyk, Shinydarkrai94, Hydriz, Pearts, Ldo, SgtBursk, Cubicsphere, Gqfreeman, Mudman65, Animajosser and Anonymous: 76
- **Blender 3D: Noob to Pro/Model a Silver Goblet cylinder** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Model_a_Silver_Goblet_cylinder?oldid=3263361 *Contributors:* Nanobug, Dooglus, Wynnmee, MSK61, Ramac, MaSabre, Adrignola, Arch dude, Rwxrwxrwx, Pearts, Ldo, LlamaAl, MercedMike, Animajosser, PokestarFanBot, Bobhaspantz and Anonymous: 16
- **Blender 3D: Noob to Pro/Spin a goblet** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Spin_a_goblet?oldid=3184967 *Contributors:* Xania, Adrignola, Arch dude, Rwxrwxrwx, Avicennasis, Ldo, Animajosser, LightspeedLife and Anonymous: 6

- **Blender 3D: Noob to Pro/Light a Silver Goblet** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Light_a_Silver_Goblet?oldid=3185098 *Contributors:* Nanobug, AverieZen, Brimman, Bullercruz1, QuiteUnusual, Adrignola, Mabdul, Arch dude, Raven-Whitehawk, Bonapon, Rwxrwxrxw, Avicennasis, WithAlligators, Tsduv21, Pearts, Ldo, Hash-enwikibooks, Mooroon, Leaderboard, MercedMike, Elfomnidnight, Taabishm2, Animajosser and Anonymous: 22
- **Blender 3D: Noob to Pro/Simple Vehicle** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Simple_Vehicle?oldid=2994156 *Contributors:* Spiderworm, ZeroOne, Popski, Derbeth, Oracleoftruth-enwikibooks, Mrfelis, AM088, Withinfocus, Sean.flynn, SchmS, Jguk, Reptiles extintos, Pengo, Dooglus, Mdd4696, AverieZen, Adrignola, Sesu Prime, Pearts, Animajosser and Anonymous: 30
- **Blender 3D: Noob to Pro/Simple Vehicle: Wheel** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Simple_Vehicle%3A_Wheel_tutorial_1?oldid=3233821 *Contributors:* Nielmo, Spiderworm, Jomegat, Tannersf, Xania, Nerdrabidmonkey, AverieZen, Feureau-enwikibooks, MSK61, Eyeonus, Chelsean528, Ma5abre, ThomasTenCate, Richard 27, CodeSnorter, IqAndreas, Rpyle731, QuiteUnusual, Argav, Adrignola, RavenWhitehawk, Xander98989, Fidoda-enwikibooks, Pearts, Ldo, JamesCrook, Iamslite, Aldnonymous, JamesNZ, Slayerjp, Kusogaki, Glaisher, JeremyS~enwikibooks, MasterSkrat, Vrazdan1000, MercedMike, Tank22119, JonnyJack, Oparisy, Brimbard, Fernando2812l, Animajosser, Rdhill734455, Bobhaspants, Ecodragon, Deepnandi619 and Anonymous: 114
- **Blender 3D: Noob to Pro/Simple Vehicle: MudTires** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Simple_Vehicle%3A_Wheel_tutorial_2?oldid=3190292 *Contributors:* Rwxrwxrxw, Aldnonymous, Buttflyhoney, LesleyLai, Elfomnidnight, Animajosser, Rdhill734455 and Anonymous: 10
- **Blender 3D: Noob to Pro/Simple Vehicle: Seat** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Simple_Vehicle%3A_Seat?oldid=3112061 *Contributors:* Spiderworm, Jomegat, Xania, AverieZen, Feureau-enwikibooks, Outbacksam34, Josellis, MSK61, Carpetsmoker, Odalcer, JackPotte, Rpyle731, QuiteUnusual, Yoshi1476, Adrignola, Arch dude, Tegel, Avicennasis, Reyk, Pearts, Ldo, Omnipgeek6, Dspaun, Slayerjp, Defender, Adam.Elenktik, Leaderboard, Animajosser, IsraeIFD and Anonymous: 32
- **Blender 3D: Noob to Pro/Simple Vehicle: Rocket Launcher** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Simple_Vehicle%3A_Rocket_Launcher?oldid=3176077 *Contributors:* Spiderworm, AverieZen, Feureau-enwikibooks, TBOL3, Mecalith-enwikibooks, Eruantalon, Recent Runes, Odalcer, KentTong, Ramac, IqAndreas, Mgjv~enwikibooks, Biomeksensei, Yoshi1476, Adrignola, NODJKQR, Sephlaire, Reyk, Pearts, Dspaun, Kekie1, MasterSkrat, MercedMike, Animajosser and Anonymous: 31
- **Blender 3D: Noob to Pro/Simple Vehicle: Body** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Simple_Vehicle%3A_Body?oldid=3120263 *Contributors:* Spiderworm, Derbeth, Xania, Beuc, AverieZen, Alexwill84, Inventor-enwikibooks, Luke321321, Newbie-enwikibooks, Samnsparky~enwikibooks, Ramac, Falconx1096, JCrawford, QuiteUnusual, Spellmaker, Adrignola, PaulJC, RavenWhitehawk, Henrybissonnette, Avicennasis, Yesuto15, Reyk, Pearts, Ldo, Aggressor, Dspaun, Kekie1, SgtBursk, Mooroon, BethNaught, MercedMike, Elfomnidnight, Animajosser and Anonymous: 35
- **Blender 3D: Noob to Pro/Simple Vehicle: Some Assembly Required** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Simple_Vehicle%3A_Some_Assembly_Required?oldid=3220600 *Contributors:* Spiderworm, AverieZen, Ascen, Radialronnie, Falconx1096, 0o0oo0k, Rpyle731, Mgjv~enwikibooks, Adrignola, Heinrich5991, Telmer6, DDD3x, Reyk, Rea00cy, Pearts, JackBot, Mooroon, MercedMike, Animajosser and Anonymous: 18
- **Blender 3D: Noob to Pro/Modeling a 3D Parachute in Blender** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Modeling_a_3D_Parachute_in_Blower?oldid=3109379 *Contributors:* Radialronnie, Null Point, Daworm, Avicennasis, Yesuto15, Dagonite, Pearts, Ldo, Kekie1, Kusogaki, Adam.Elenktik and Anonymous: 17
- **Blender 3D: Noob to Pro/Model a Low Poly Head** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Model_a_Low_Poly_Head?oldid=2992135 *Contributors:* Jomegat, Jguk, Sumoncil, Wynnm, Tommciver, Beuc, Rm5248, DEAFCODE, Ravichandar84, MSK61, Radialronnie, Kayau, YMS, Adrignola, Sandsmertz, Daworm, Reyk, Pearts, Ldo, Kekie1, JadenHorst, Broner, Animajosser and Anonymous: 24
- **Blender 3D: Noob to Pro/Building a House** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Building_a_House?oldid=3042363 *Contributors:* Wynnm, Az1568, Beuc, Reece, MSK61, Slush15player, Recent Runes, POpZ, Waximo, Ramac, Neilwright-man~enwikibooks, SoylentGreen, Bullercruz1, Bruce89, YMS, QuiteUnusual, Adrignola, Brandished, Sandsmertz, CFrey, Avicennasis, Mast3rlinx, Pavroo, Pearts, Ldo, MercedMike, LastFable, Animajosser and Anonymous: 42
- **Blender 3D: Noob to Pro/Pipe joints** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Pipe_joints?oldid=3084778 *Contributors:* Panic2k4, Beuc, SoylentGreen, Adrignola, Begeun, Camp, Vaakmeisster, WithAlligators, Tal500, Pearts, Ldo, MarkLorenWilson, ~riley, Marr Gilb, MercedMike, Anarchistamy, Animajosser, Rasmustrew and Anonymous: 16
- **Blender 3D: Noob to Pro/Lighting Suzanne: Introductory one lamp lighting** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Lighting_Suzanne%3A_Introductory_one_lamp_lighting?oldid=2992347 *Contributors:* MercedMike and Animajosser
- **Blender 3D: Noob to Pro/Curve and Path Modeling** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Curve_and_Path_Modeling?oldid=2751405 *Contributors:* Panic2k4, Spiderworm, Zoohouse, Popski, Tmandry~enwikibooks, Lazy-lump, SergeantOreo, Atallcostsky, Bullercruz1, Adrignola, Pearts, Ldo, MercedMike and Anonymous: 9
- **Blender 3D: Noob to Pro/Intro to Bezier Curves** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Intro_to_Bezier_Curves?oldid=3197609 *Contributors:* Nanobug, DavidCary, Lazy-lump, Atallcostsky, IqAndreas, Neurotic~enwikibooks, Spellmaker, Adrignola, Avicennasis, Reyk, Pearts, Ldo, Marr Gilb, SgtBursk, Vogone, MercedMike, Animajosser, Bobhaspants and Anonymous: 25
- **Blender 3D: Noob to Pro/Beveling a Curve** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Beveling_a_Curve?oldid=3064620 *Contributors:* Ldo, Anarchistamy and Win Bigly
- **Blender 3D: Noob to Pro/NURBS Patches** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/NURBS_Patches?oldid=2753611 *Contributors:* Ldo and Anonymous: 1
- **Blender 3D: Noob to Pro/Deforming Meshes using the Curve Modifier** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Deforming_Meshes_using_the_Curve_Modifier?oldid=3197618 *Contributors:* Jbalint~enwikibooks, Lazy-lump, Mike.lifeguard, Embri, WiTr, Bullercruz1, Null Point, Encesil, Avicennasis, Reyk, Pearts, Ldo, Aggressor, MercedMike, Animajosser, Bobhaspants and Anonymous: 19

- **Blender 3D: Noob to Pro/The Empty Object** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/The_Empty_Object?oldid=2985246 *Contributors:* Ldo, MercedMike, Anarchistamy and Animajosser
- **Blender 3D: Noob to Pro/Background Images** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Background_Images?oldid=2755534 *Contributors:* Ldo, MasterSkrat and Anonymous: 1
- **Blender 3D: Noob to Pro/Aligning Vertices with a Guide Image** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Aligning_Vertices_with_a_Guide_Image?oldid=3176307 *Contributors:* Arch dude, Avicennasis, Pearts, Ldo, Stepheng3, HethrirBot, Kekie1, Erops19, AtheneNoctua, MercedMike, ZarabethLanger, Animajosser and Anonymous: 10
- **Blender 3D: Noob to Pro/Modeling a Fox from Guide Images** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Modeling_a_Wolf_from_Guide_Images?oldid=3239372 *Contributors:* Spiderworm, DuLithgow, AM088, Stormy, Random9q, Jguk, Hagindaz, Dmwick, Nczempin, Mddd4696, Norleaf, Paperclip~enwikibooks, Badalia, Goatofdeath, Beuc, Herractic, Moreejt, AverieZen, Lebigot, Mudpuddle, Longsword~enwikibooks, Twofingers, Lirtferk, Yewbow, MSK61, Stefantalpalaru, Pedro Fonini, Moohasha, Tyaedalis, Ramac, SoylentGreen, Bullercruz1, JackPotte, YMS, QuiteUnusual, Sigma 7, Adrignola, Arch dude, Zilch~enwikibooks, Null Point, Alpha0~enwikibooks, Efren~enwikibooks, Sandsmertz, Camp, Avicennasis, Skatanist, Pearts, Ldo, Bcroner, MercedMike, Fernando2812l, Animajosser, Charlesmmoye and Anonymous: 117
- **Blender 3D: Noob to Pro/2D Image (logo) to a 3D Model** *Source:* [https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/2D_Image_\(logo\)_to_a_3D_Model?oldid=3099552](https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/2D_Image_(logo)_to_a_3D_Model?oldid=3099552) *Contributors:* Nanobug, Sunny256, GeocacherNemesis, Iamunknow, Spiderworm, Jshadias~enwikibooks, Popski, Derbeth, Allefant~enwikibooks, Mrfelis, Danielsimlind, Sterre~enwikibooks, Darkonc, Shokuku, SchmS, Jguk, Johno, Jawboot, Kieran~enwikibooks, Shadowimage, Goatofdeath, Az1568, Tannersf, Twofingers, Lazy-lump, Commons-Delinker, Yoyotweak, Mike.lifeguard, Astharoth~enwikibooks, Gumanoe, Yiyyas, Turtle Man, Bullercruz1, Spellmaker, Adrignola, Thor-rack~enwikibooks, Null Point, Avicennasis, Pearts, Ldo, Marr Gilb, Velichi, VB4231, Anarchistamy, Animajosser and Anonymous: 99
- **Blender 3D: Noob to Pro/Subsurface scattering** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Subsurface_scattering?oldid=2561641 *Contributors:* Ldo
- **Blender 3D: Noob to Pro/Ray Tracing** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Ray_Tracing?oldid=2650059 *Contributors:* Ldo and Anarchistamy
- **Blender 3D: Noob to Pro/Using Textures** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Using_Textures?oldid=3208829 *Contributors:* Beuc, SoylentGreen, JackPotte, Adrignola, Mattiasl, Avicennasis, Ldo, Neverbirth, Kusogaki, Animajosser and Anonymous: 14
- **Blender 3D: Noob to Pro/Using a texture to make a material partially transparent** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Using_a_texture_to_make_a_material_partially_transparent?oldid=3006884 *Contributors:* Beuc, SoylentGreen, Adrignola, Robfox70, Avicennasis, WithAlligators, Ldo, Weeix, Kusogaki, Lindlawlet and Anonymous: 5
- **Blender 3D: Noob to Pro/Creating Basic Seawater** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Creating_Basic_Seawater?oldid=3148691 *Contributors:* Spiderworm, Popski, AM088, Sterre~enwikibooks, Khan~enwikibooks, Iluvblender, SchmS, Accurrent~enwikibooks, DanMan, Tannersf, SoylentGreen, Bullercruz1, Adrignola, Daworm, Dobz, Avicennasis, Ldo, Weeix, Cindy Blendy, MercedMike and Anonymous: 15
- **Blender 3D: Noob to Pro/Mountains Out Of Molehills 2** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Mountains_Out_Of_Molehills_2?oldid=2990152 *Contributors:* Spiderworm, AM088, SchmS, BeBraw, AverieZen, Ask~enwikibooks, Arcticfreeze83, Bullercruz1, Adrignola, Sendoshin, Dobz, WindsorSpring, Animajosser and Anonymous: 7
- **Blender 3D: Noob to Pro/Basic Carpet Texture** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Basic_Carpet_Texture?oldid=2990153 *Contributors:* Leinad13, SchmS, AverieZen, Dan13, Mochimo, SergeantOreo, Bullercruz1, Nick.anderegg, Adrignola, Bonapon, Avicennasis, Krismania, Ldo, Weeix, Animajosser and Anonymous: 24
- **Blender 3D: Noob to Pro/The Rusty Ball** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/The_Rusty_Ball?oldid=3157426 *Contributors:* Argento, Spiderworm, Soriyath, AM088, SchmS, DanMan, Mochimo, Brylie, Bullercruz1, Adrignola, Baadtaste, WithAlligators, Ownthor, Ldo, Weeix, Blendegeek, MercedMike, Animajosser, LightspeedLife and Anonymous: 14
- **Blender 3D: Noob to Pro/Creating Pixar-looking eyes in Blender** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Creating_Pixar-looking_eyes_in_Blender?oldid=2996095 *Contributors:* Furrykef, Yosofun~enwikibooks, Jleedev, WeirdHat, Jguk, Pengo, DanMan, Revertentative, Harkin, Bach~enwikibooks, Badalia, Chinmay gautam, Az1568, Tannersf, Rdp, Philipmac, Hrakaroo, Mochimo, Mike.lifeguard, Bmarnz, ThomasB~enwikibooks, Icky-wiki, SergeantOreo, Thud~enwikibooks, Davidwilliams, Kylstoman, Bullercruz1, QuiteUnusual, Adrignola, Null Point, PKHG, Irakrakow, Henrybissonnette, WithAlligators, Ldo, MercedMike, Jawelik and Anonymous: 60
- **Blender 3D: Noob to Pro/Procedural Eyeball** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Procedural_Eyeball?oldid=3177804 *Contributors:* Jmartens, Hagindaz, Jorge Morais, Radialronnie, Kayau, Bullercruz1, Tdunn7, Spellmaker, Lordelph~enwikibooks, Wolveenmoon, Marinalan, 00Anme00, Bonapon, Avicennasis, Mast3rlinx, WithAlligators, Ldo, Kekie1, Kusogaki, MercedMike, Animajosser, Chloris pale green and Anonymous: 23
- **Blender 3D: Noob to Pro/Putting It All Together: A Dragon!** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Putting_It_All_Together%3A_A_Dragon!?oldid=3159142 *Contributors:* MercedMike, Animajosser and LightspeedLife
- **Blender 3D: Noob to Pro/Image Editor** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Image_Editor?oldid=2657342 *Contributors:* Ldo, MercedMike and Anarchistamy
- **Blender 3D: Noob to Pro/UV Map Basics** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/UV_Map_Basics?oldid=3139277 *Contributors:* Argento, Spiderworm, ZeroOne, Jonon~enwikibooks, Popski, Mico~enwikibooks, Arru~enwikibooks, SchmS, LordOfer, Az1568, Nowic, Jamin3D~enwikibooks, TBOL3, LokiClock, Damian Yerrick, Remi, SergeantOreo, Atallcostsky, Jeff G, Anonymous Dissident, Mohax, SoylentGreen, Arcticfreeze83, Bullercruz1, YMS, QuiteUnusual, Adrignola, Null Point, Thespeedylife, HHBones, Irakrakow, Sestina~enwikibooks, DDD3x, Avicennasis, Ldo, WindsorSpring, SiPlus, Jonno333, Othmanskn2, JamesNZ, VB4231, Leaderboard, Anarchistamy, Animajosser, Sirusdark 2016 and Anonymous: 97
- **Blender 3D: Noob to Pro/Realistic Eyes In Blender** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Realistic_Eyes_In_Blender?oldid=3120038 *Contributors:* King of Hearts, Quantum Anomaly, Bullercruz1, Nick chang207, QuiteUnusual, Null Point, ScaroDj, Ldo, Wackyvorlon, Animajosser and Anonymous: 11

- **Blender 3D: Noob to Pro/Beginning Lighting** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/BEGINNING_Lighting?oldid=2527430 *Contributors:* Nanobug, Spiderworm, Crouch, Popski, ParallaxTZ, Beuc, Bullercruz1, Adrignola, Ldo and Anonymous: 4
- **Blender 3D: Noob to Pro/Understanding Real Lights** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Understanding_Real_Lights?oldid=2562230 *Contributors:* Jomegat, Beuc, Adrignola, Tyrant monkey, Ldo, Cubicsphere and Anonymous: 5
- **Blender 3D: Noob to Pro/Understanding Blender Lights** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Understanding_Blender_Lights?oldid=2836750 *Contributors:* Ldo, MercedMike and Anonymous: 1
- **Blender 3D: Noob to Pro/Basic Lighting Rigs** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Basic_Lighting_Rigs?oldid=3172142 *Contributors:* Ldo, MasterSkrat and StevenMock
- **Blender 3D: Noob to Pro/Faked Gi with Blender internal** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Faked_Gi_with_Blender_internal?oldid=3177886 *Contributors:* Radialronnie, Adrignola, Orfo~enwikibooks, Null Point, MidnightLightning, Ldo, Cubicsphere, Animajosser, Chloris pale green and Anonymous: 8
- **Blender 3D: Noob to Pro/Parenting** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Parenting?oldid=3178008 *Contributors:* Ldo, MasterSkrat, Chloris pale green and Anonymous: 2
- **Blender 3D: Noob to Pro/Basic Animation** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Basic_Animation?oldid=3041649 *Contributors:* Crouch, Popski, Jclee, EatMyShortz~enwikibooks, ParallaxTZ, Jomegat, Peteturte~enwikibooks, AndyD, Xania, Rm5248, TBOL3, SergeantOreo, Adrignola, Tegel, 492star, Ldo and Anonymous: 26
- **Blender 3D: Noob to Pro/Basic Animation/Keyframing Introduction** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Basic_Animation/Keyframing_Introduction?oldid=3022328 *Contributors:* Ldo, Animajosser and Anonymous: 1
- **Blender 3D: Noob to Pro/The Ways of the Animator** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/The_Ways_of_the_Animator?oldid=2695361 *Contributors:* Ldo and MasterSkrat
- **Blender 3D: Noob to Pro/Animation Editors** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Animation_Editors?oldid=2695362 *Contributors:* Ldo and MasterSkrat
- **Blender 3D: Noob to Pro/Basic Animation/Introducing the Graph Editor** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Basic_Animation/Introducing_the_Graph_Editor?oldid=3038912 *Contributors:* Ldo, MasterSkrat and Animajosser
- **Blender 3D: Noob to Pro/Basic Animation/Rendering** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Basic_Animation/Rendering?oldid=2647219 *Contributors:* Ldo and Anonymous: 1
- **Blender 3D: Noob to Pro/Basic Animation/Lattice** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Basic_Animation/Lattice?oldid=3178469 *Contributors:* Jomegat, Jguk, Peteturte~enwikibooks, AndyD, Rm5248, CommonsDelinker, Brinman, Jackechan, Adrignola, Avicennasis, Kenkaku, Ldo, Gaandolf, Kusogaki, Chloris pale green and Anonymous: 28
- **Blender 3D: Noob to Pro/Basic Animation/Bounce** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Basic_Animation/Bounce?oldid=3042948 *Contributors:* Jomegat, Jguk, Mstram, AndyD, Teddygaye, Xania, Herbythyme, TBOL3, Miika22, Mike.lifeguard, Brinman, Recent Runes, AndrewBuck, Malaz, Adrignola, Avicennasis, Lionelb, Slashme, Ldo, Lowbot0, MercedMike and Anonymous: 33
- **Blender 3D: Noob to Pro/Creating Basic Water animation** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Creating_Basic_Water_animation?oldid=3195853 *Contributors:* Jomegat, Zeviion, SergeantOreo, Ramac, AndrewBuck, Malaz, Jon-comp12~enwikibooks, Vaxine19, Soeb, QuiteUnusual, Nick.anderegg, Null Point, Crystal Visions, Bonapon, Woodstocket, Avicennasis, Slashme, Mindbulletmatrix, Othmanskn, RandomDSplayer, Tropicalkitty, Animajosser, Chloris pale green and Anonymous: 41
- **Blender 3D: Noob to Pro/Flying Through A Canyon** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Flying_Through_A_Canyon?oldid=3053662 *Contributors:* GeniXPro, Funky weasel, Psychogears~enwikibooks, AndrewBuck, Tosnic, Arcticfreeze83, Null Point, Dogwynn, Bonapon, Electus~enwikibooks, Avicennasis, WithAlligators, Looterguf, Slashme, Ldo, Othmanskn2, JamesNZ, Drblitzkrieg, Animajosser and Anonymous: 17
- **Blender 3D: Noob to Pro/Using the sequencer to compile frames into an animation** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Using_the_sequencer_to_compile_frames_into_an_animation?oldid=3053732 *Contributors:* Mike.lifeguard, Radialronnie, Null Point, Professional noob, WithAlligators, Ldo and Animajosser
- **Blender 3D: Noob to Pro/Further Rendering Options** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Further_Rendering_Options?oldid=3053749 *Contributors:* Ldo, MasterSkrat and Animajosser
- **Blender 3D: Noob to Pro/Particle Systems** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Particle_Systems?oldid=3164792 *Contributors:* ParallaxTZ, Jguk, Funky weasel, AndyD, Romainbehar, TBOL3, Danthemango, SergeantOreo, Soylent-Green, Adrignola, Null Point, MarkM~enwikibooks, Ldo, Animajosser, LightspeedLife and Anonymous: 17
- **Blender 3D: Noob to Pro/Making Fire** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Making_Fire?oldid=3108402 *Contributors:* Panic2k4, Derbeth, Chw333, ParallaxTZ, Jomegat, Jguk, Mstram, Funky weasel, AndyD, Teddygaye, Chinmay gautam, Romainbehar, Michaelsobers, Venar303, Dimzy Doodle, SoylentGreen, QuiteUnusual, Adrignola, Electus~enwikibooks, Ldo, Lordscales91, Bcroner, BethNaught and Anonymous: 44
- **Blender 3D: Noob to Pro/Furry** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Furry?oldid=3056593 *Contributors:* Beuc, SoylentGreen, Adrignola, ScaroDj, WithAlligators, 492star, Looterguf, Ldo, MercedMike, Animajosser and Anonymous: 12
- **Blender 3D: Noob to Pro/Fireworks** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Fireworks?oldid=3043158 *Contributors:* ZeroOne, MrMagee2759~enwikibooks, AndrewBuck, Jeff G., SoylentGreen, Kayau, 00Anme00, Digitalcircuit, Der Künstler, Ldo, Lindlawlet and Anonymous: 8
- **Blender 3D: Noob to Pro/Particles forming Shapes** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Particles_forming_Shapes?oldid=3057208 *Contributors:* SoylentGreen, Avicennasis, WithAlligators, Looterguf, Ldo, Lordscales91, Lindlawlet, Animajosser and Anonymous: 3

- **Blender 3D: Noob to Pro/Billboard Animation** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Billboard_Animation?oldid=3057871 *Contributors:* SoylentGreen, Avicennasis, Ldo, Animajosser and Anonymous: 1
- **Blender 3D: Noob to Pro/Soft Body Animation** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Soft_Body_Animation?oldid=3057918 *Contributors:* Argento, ZeroOne, Jguk, Porlando, Venar303, Atallcostsky, AndrewBuck, Tosnic, SoylentGreen, Arcticfreeze83, Adrignola, Avicennasis, WithAlligators, Ldo, Intityx, Stepheng3, Animajosser and Anonymous: 17
- **Blender 3D: Noob to Pro/Simple Cloth Animation** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Simple_Cloth_Animation?oldid=3179603 *Contributors:* Argento, Super3boy, Jguk, Hagindaz, Scottpledger, Bluincher, Venar303, Selfhiding, SergeantOreo, Malaz, Tosnic, That.1.guy, Adrignola, Null Point, CwadrupldijitGoldilox, Ldo, Chloris pale green and Anonymous: 24
- **Blender 3D: Noob to Pro/Soft Body with wind** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Soft_Body_with_wind?oldid=3063521 *Contributors:* ZeroOne, Eb264, QuiteUnusual, Rukaru, Null Point, Sendoshin, Animajosser and Anonymous: 9
- **Blender 3D: Noob to Pro/Your First Test** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Your_First_Test?oldid=3179604 *Contributors:* Spiderworm, Allefant~enwikibooks, Jomegat, Hammurderer, Malaz, 5o~enwikibooks, Adrignola, Null Point, Othmanskn2, Animajosser, Chloris pale green and Anonymous: 21
- **Blender 3D: Noob to Pro/Platformer: Creation and Controls** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro_Platformer%3A_Creation_and_Controls?oldid=3081365 *Contributors:* CommonsDelinker, LokiClock, Neoptolemus, Malaz, Mhoram, Ig92298, Animajosser, Smileyarc and Anonymous: 4
- **Blender 3D: Noob to Pro/An aMAZEing game engine tutorial** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro_An_aMAZEing_game_engine_tutorial?oldid=3063524 *Contributors:* Jomegat, Thenub314, LokiClock, Malaz, Seraphrevan~enwikibooks, Thud~enwikibooks, Kylstoman, Turtle Man, AttilaMH, Sendoshin, Electus~enwikibooks, Avicennasis, Sankekur, Boywonder~enwikibooks, Harrybrowne1986, Sym90, Lindlawlet, Animajosser and Anonymous: 41
- **Blender 3D: Noob to Pro/Platformer: Physics Fixes** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro_Platformer%3A_Physics_Fixes?oldid=3063526 *Contributors:* LokiClock, Malaz, QuiteUnusual, Null Point, Avicennasis, Animajosser and Anonymous: 3
- **Blender 3D: Noob to Pro/Making exe** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Making_exe?oldid=3063527 *Contributors:* Derbeth, Jguk, Freakazo v2.1, Venar303, Chelseafan528, Malaz, Kylstoman, Turtle Man, Arcticfreeze83, Muimi~enwikibooks, AttilaMH, GeForce3, Jackred, Dogwynn, Avicennasis, DeltaSpeeds, Skirrid, JsbrownDog, Mavery1, RandomD-Splayer, Animajosser and Anonymous: 26
- **Blender 3D: Noob to Pro/Build a skybox** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Build_a_skybox?oldid=3220571 *Contributors:* Allefant~enwikibooks, Greensweater, Jguk, Glome~enwikibooks, Malaz, AttilaMH, Adrignola, Null Point, Irakrakow, Avicennasis, JackBot, Animajosser and Anonymous: 8
- **Blender 3D: Noob to Pro/Basic mouse pointer** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Basic_mouse_pointer?oldid=3063528 *Contributors:* Jguk, Freakazo v2.1, Justthisguy, Atallcostsky, Malaz, MrThreepwood, AttilaMH, Adrignola, Null Point, Animajosser and Anonymous: 4
- **Blender 3D: Noob to Pro/Text in BGE** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Text_in_BGE?oldid=3063520 *Contributors:* Chuckhoffmann, Bullercruz1, AttilaMH, Adrignola, Null Point, Animajosser and Anonymous: 3
- **Blender 3D: Noob to Pro/Python Platformer: Creation** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro_Python_Platformer%3A_Creation?oldid=3063530 *Contributors:* LokiClock, Turtle Man, Null Point, Brimbard and Animajosser
- **Blender 3D: Noob to Pro/Advanced Tutorials/Python Scripting/Introduction** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro_Advanced_Tutorials/Python_Scripting/Introduction?oldid=3111023 *Contributors:* Argento, Allefant~enwikibooks, Ideasman42, Jguk, Turtle Man, Xlinuxkdex, Ggolem1, Adrignola, Robstafarian, Ftiercel, Justlost, Admirj, Ldo and Anonymous: 16
- **Blender 3D: Noob to Pro/Advanced Tutorials/Python Scripting/Addon Anatomy** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro_Advanced_Tutorials/Python_Scripting/Addon_Anatomy?oldid=3071827 *Contributors:* Ldo, Rant~enwikibooks, Ron333 and Anonymous: 3
- **Blender 3D: Noob to Pro/Advanced Tutorials/Python Scripting/Addon User Interface** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro_Advanced_Tutorials/Python_Scripting/Addon_User_Interface?oldid=3071828 *Contributors:* Ldo, Bugaevc, Rant~enwikibooks and Anonymous: 3
- **Blender 3D: Noob to Pro/Advanced Tutorials/Python Scripting/Addon Custom Property** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro_Advanced_Tutorials/Python_Scripting/Addon_Custom_Property?oldid=3109911 *Contributors:* Ldo, Animajosser and Anonymous: 1
- **Blender 3D: Noob to Pro/Advanced Tutorials/Python Scripting/Separately Installable Addon** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro_Advanced_Tutorials/Python_Scripting/Separately_Installable_Addon?oldid=3071830 *Contributors:* Ldo and Animajosser
- **Blender 3D: Noob to Pro/Advanced Tutorials/Python Scripting/Object, Action, Settings** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro_Advanced_Tutorials/Python_Scripting/Object%2C_Action%2C_Settings?oldid=3071832 *Contributors:* Ldo, Brimbard, Animajosser and Anonymous: 1
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Modeling** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro_Advanced_Tutorials/Advanced_Modeling?oldid=3179607 *Contributors:* Turtle Man, Bullercruz1, Adrignola, Ldo, Chloris pale green and Anonymous: 1
- **Blender 3D: Noob to Pro/HDRi** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro_HDRi?oldid=3072550 *Contributors:* Argento, Dan-aka-jack, Bmud, ErikBongers, Chelseafan528, Turtle Man, Bullercruz1, QuiteUnusual, Adrignola, Maebe, Avicennasis, Jinincarnate and Anonymous: 14
- **Blender 3D: Noob to Pro/Creating a Light Probe** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro_Creating_a_Light_Probe?oldid=3112655 *Contributors:* Bmud, Allefant~enwikibooks, Recent Runes, Bullercruz1, Nick.anderegg, Avicennasis, Animajosser and Anonymous: 5

- **Blender 3D: Noob to Pro/Making Landscapes with heightmaps** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Making_Landscapes_with_heightmaps?oldid=3072562 *Contributors:* Xerol, Jguk, Aidan.Sullivan, Sandalle~enwikibooks, Lazy-lump, Pedro Fonini, Ekrickyote, Chelseafan528, Alex L33, Kirona, Ascen, Wikidieter, Demoathon, Jackechan, Root~enwikibooks, Johnstarbird, UnknownGuy, ShadoCrytr~enwikibooks, Noob sks dk, CodeSnorter, Lorhc~enwikibooks, Bullercruz1, QuiteUnusual, Ubai~enwikibooks, Adrignola, Pkramer509, Animajosser and Anonymous: 42
- **Blender 3D: Noob to Pro/How to Do Procedural Landscape Modeling** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/How_to_Do_Procedural_Landscape_Modeling?oldid=3073000 *Contributors:* SergeantOreo, Pedro Fonini, Moohasha, Sketlover, AndrewBuck, Roli~enwikibooks, Ma5abre, Oasiac, Lindlawlet, Animajosser and Anonymous: 17
- **Blender 3D: Noob to Pro/Landscape Modeling I: Basic Terrain** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Landscape_Modeling_I%3A_Basic_Terrain?oldid=3073455 *Contributors:* Billymac00, Moohasha, Rekov, KentTong, Blenderman~enwikibooks, AndrewBuck, Root~enwikibooks, QuiteUnusual, Zilla~enwikibooks, Adrignola, Kindeller, Avicennasis, RaymondSutanto, Syum90, Animajosser and Anonymous: 35
- **Blender 3D: Noob to Pro/Landscape Modeling II: Texture Stenciling** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Landscape_Modeling_II%3A_Texture_Stenciling?oldid=3073025 *Contributors:* Mike.lifeguard, Moohasha, UnknownGuy, Adrignola, Avicennasis, Bzarnal and Anonymous: 20
- **Blender 3D: Noob to Pro/Landscape Modeling III: Exporting as a Heightmap** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Landscape_Modeling_III%3A_Exporting_as_a_Heightmap?oldid=3073454 *Contributors:* Moohasha, Bullercruz1, Ubai~enwikibooks, Adrignola, Avicennasis, Ldo, Animajosser and Anonymous: 8
- **Blender 3D: Noob to Pro/Bump Mapping** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Bump_Mapping?oldid=2525729 *Contributors:* SoylentGreen, Turtle Man, Bullercruz1, Adrignola, Ldo and Anonymous: 8
- **Blender 3D: Noob to Pro/Normal Mapping** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Normal_Mapping?oldid=2498437 *Contributors:* SoylentGreen, Turtle Man, Bullercruz1, Nick.anderegg, Adrignola and Anonymous: 1
- **Blender 3D: Noob to Pro/Texture Normal Mapping** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Texture_Normal_Mapping?oldid=3073460 *Contributors:* SoylentGreen, Bullercruz1, Nick.anderegg, Adrignola, Animajosser and Anonymous: 1
- **Blender 3D: Noob to Pro/Color Map Normal Mapping** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Color_Map_Normal_Mapping?oldid=3073461 *Contributors:* SoylentGreen, Bullercruz1, Markharper80, Adrignola, Dudekahedron, Avicennasis, WithAlligators, Animajosser and Anonymous: 3
- **Blender 3D: Noob to Pro/Nodes** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Nodes?oldid=2650124 *Contributors:* Turtle Man, Adrignola, Orindlincoln, Irakrakow, Ldo and Anonymous: 1
- **Blender 3D: Noob to Pro/Texture Nodes** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Texture_Nodes?oldid=2525084 *Contributors:* Ldo, Stepheng3 and Anonymous: 1
- **Blender 3D: Noob to Pro/Material Nodes** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Material_Nodes?oldid=3100336 *Contributors:* Turtle Man, Avicennasis, Ldo and Animajosser
- **Blender 3D: Noob to Pro/Compositing** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Compositing?oldid=3063512 *Contributors:* Turtle Man, Ldo, Win Bigly and Anonymous: 1
- **Blender 3D: Noob to Pro/Compositing/Portal Effect** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Compositing/Portal_Effect?oldid=3100367 *Contributors:* Ldo and Animajosser
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Rendering** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Rendering?oldid=2650130 *Contributors:* Ldo and Anonymous: 1
- **Blender 3D: Noob to Pro/Advanced Tutorials/Introduction to Cycles** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Introduction_to_Cycles?oldid=3100389 *Contributors:* Ldo and Animajosser
- **Blender 3D: Noob to Pro/Advanced Tutorials/Cycles Glass** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Cycles_Glass?oldid=3100633 *Contributors:* Ldo and Animajosser
- **Blender 3D: Noob to Pro/Advanced Tutorials/Cycles Fireflies** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Cycles_Fireflies?oldid=3100634 *Contributors:* Ldo, Animajosser and Anonymous: 1
- **Blender 3D: Noob to Pro/Advanced Tutorials/Cycles Fireflies 2** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Cycles_Fireflies_2?oldid=3100716 *Contributors:* Ldo and Animajosser
- **Blender 3D: Noob to Pro/Advanced Tutorials/Procedural Eyeball in Cycles** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Procedural_Eyeball_in_Cycles?oldid=3100632 *Contributors:* Ldo and Animajosser
- **Blender 3D: Noob to Pro/Advanced Tutorials/Introduction to Freestyle** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Introduction_to_Freestyle?oldid=2645234 *Contributors:* Ldo
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation?oldid=3101939 *Contributors:* Gabio, Jguk, Turtle Man, Bullercruz1, Adrignola, Ldo, Animajosser and Anonymous: 7
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Introduction** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Introduction?oldid=3100977 *Contributors:* Gabio, Pamtango, Adhemar~enwikibooks, Jguk, Zachary Murray~enwikibooks, Thinkingman~enwikibooks, Turtle Man, Adrignola, Animajosser and Anonymous: 4
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/index** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/index?oldid=2094098 *Contributors:* Gabio, Adhemar~enwikibooks, Jguk, Turtle Man, Adrignola and Anonymous: 1
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Armature/index** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Armature/index?oldid=2288250 *Contributors:* Gabio, Pamtango, Adhemar~enwikibooks, Jguk, Adrignola, Nkansahrexford and Anonymous: 9

- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Armature/object** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Armature/object?oldid=3101015 *Contributors:* Allefant~enwikibooks, Jmartens, Gabio, Pamtango, Super3boy, Jguk, Turtle Man, Adrignola, Avicennasis, Nkansahrexford, Animajosser and Anonymous: 12
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Armature/edit** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Armature/edit?oldid=3101040 *Contributors:* Allefant~enwikibooks, Soylentgreen, Gabio, Madcello, Jguk, LokiClock, Turtle Man, QuiteUnusual, Adrignola, Animajosser and Anonymous: 23
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Armature/pose** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Armature/pose?oldid=3101042 *Contributors:* Soylentgreen, Gabio, Jguk, Turtle Man, Adrignola, Animajosser and Anonymous: 13
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Mesh/index** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Mesh/index?oldid=2094094 *Contributors:* Gabio, Jguk, Turtle Man, Adrignola and Anonymous: 4
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Mesh/Amodif** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Mesh/Amodif?oldid=3101104 *Contributors:* Gabio, Jguk, Willemd, Turtle Man, Adrignola, Dreas~enwikibooks, Animajosser and Anonymous: 11
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Mesh/env** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Mesh/env?oldid=3101245 *Contributors:* Gabio, Jguk, Turtle Man, Adrignola, Henrybissonnette, Animajosser and Anonymous: 10
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Mesh/vg** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Mesh/vg?oldid=3101249 *Contributors:* Gabio, Madcello, Stalepanda, Jguk, Turtle Man, Adrignola, Avicennasis, HethrirBot, Animajosser and Anonymous: 15
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Mesh/Shape** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Mesh/Shape?oldid=3101274 *Contributors:* Gabio, Jguk, AndyD, Turtle Man, Adrignola, Animajosser and Anonymous: 5
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Mesh/Shape/Sync** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Mesh/Shape/Sync?oldid=3174112 *Contributors:* Jguk, AndyD, TBOL3, Turtle Man, QuiteUnusual, Nick.anderegg, Adrignola, Avicennasis, Animajosser and Anonymous: 20
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Const/index** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Const/index?oldid=3101336 *Contributors:* Allefant~enwikibooks, Soylentgreen, Gabio, Dipingo~enwikibooks, Jguk, Mstram, AndyD, Turtle Man, Adrignola, Animajosser and Anonymous: 6
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Const/cl** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Const/cl?oldid=3101555 *Contributors:* Derbeth, Iluvblender, Gabio, Jguk, LokiClock, Turtle Man, Adrignola, Animajosser and Anonymous: 5
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Const/cr** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Const/cr?oldid=3101556 *Contributors:* Gabio, Jguk, Turtle Man, Adrignola, Slashme, Animajosser and Anonymous: 3
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Const/tt** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Const/tt?oldid=3101558 *Contributors:* Gabio, Jguk, Webaware, Turtle Man, Adrignola, Animajosser and Anonymous: 4
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Const/f1** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Const/f1?oldid=3101646 *Contributors:* Allefant~enwikibooks, Gabio, Jguk, Turtle Man, Adrignola, Animajosser and Anonymous: 12
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Const/lt** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Const/lt?oldid=3101647 *Contributors:* Turtle Man and Animajosser
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Const/fp** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Const/fp?oldid=3101582 *Contributors:* Gabio, Jguk, Scorepion13, Turtle Man, QuiteUnusual, Adrignola, HethrirBot and Animajosser
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Const/st** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Const/st?oldid=3101594 *Contributors:* Gabio, Jguk, SergeantOreo, Turtle Man, Adrignola, Animajosser and Anonymous: 1
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/Const/ik** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/Const/ik?oldid=3101596 *Contributors:* Allefant~enwikibooks, Jomegat, Gabio, Jguk, SergeantOreo, Turtle Man, Adrignola, Animajosser and Anonymous: 15
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/timeline** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/timeline?oldid=3101651 *Contributors:* Gabio, Jguk, Turtle Man, Adrignola and Animajosser
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/ipo/index** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/ipo/index?oldid=3101652 *Contributors:* Gabio, Jguk, Turtle Man, Adrignola and Animajosser
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/ipo/type** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/ipo/type?oldid=3101659 *Contributors:* Gabio, Jguk, Turtle Man, Adrignola and Animajosser

- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/NLA/index** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/NLA/index?oldid=3101664 *Contributors:* Cspurrier, Halley, Jguk, Tannersf, Turtle Man, Adrignola, Cymru.lass, Animajosser and Anonymous: 4
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/NLA/intro** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/NLA/intro?oldid=3101724 *Contributors:* Darklama, Zeviion, Turtle Man, Adrignola, Avicennasis, Animajosser and Anonymous: 2
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/Guided tour/NLA/stride** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/Guided_tour/NLA/stride?oldid=2094086 *Contributors:* Jguk, Turtle Man, Adrignola and Anonymous: 2
- **Blender 3D: Noob to Pro/Advanced Tutorials/Relative Vertex Keys** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Relative_Vertex_Keys?oldid=2654192 *Contributors:* Popski, Allefant~enwikibooks, Quaddux, Jguk, Joystick Tutorial, Remi0o, Turtle Man, Bullercruz1, QuiteUnusual, Van der Hoorn, Adrignola and Anonymous: 11
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/example/bob/index** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/example/bob/index?oldid=3101727 *Contributors:* Wavez, Jguk, Turtle Man, Adrignola, Animajosser and Anonymous: 11
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/example/bob/Build rig** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/example/bob/Build_rig?oldid=3263356 *Contributors:* Wavez, Jguk, Turtle Man, Adrignola, Animajosser, PokestarFanBot and Anonymous: 1
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/example/bob/connect2mesh** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/example/bob/connect2mesh?oldid=3101716 *Contributors:* Jomegat, Turtle Man, Animajosser and Anonymous: 1
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/example/bob/walk** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/example/bob/walk?oldid=3101717 *Contributors:* Turtle Man, QuiteUnusual, Animajosser and Anonymous: 1
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/example/ Piston, Rod and Crank** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/example/_Piston%2C_Rod_and_Crank?oldid=3101718 *Contributors:* Darklama, Soylentgreen, Recent Runes, Turtle Man, Adrignola, Juetho, Avicennasis, Animajosser and Anonymous: 6
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Animation/example/ Cutting Through Steel** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Animation/example/_Cutting_Through_Steel?oldid=3101763 *Contributors:* SoylentGreen, Turtle Man, Adrignola, Null Point, Avicennasis, Animajosser and Anonymous: 2
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Game Engine** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Game_Engine?oldid=2150404 *Contributors:* Xerol, Turtle Man, Miyazaki~enwikibooks, Bullercruz1, Adrignola and Anonymous: 1
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Game Engine/Game Creating Techniques(GUI)** *Source:* [https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Game_Engine/Game_Creating_Techniques\(GUI\)?oldid=2094054](https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Game_Engine/Game_Creating_Techniques(GUI)?oldid=2094054) *Contributors:* Turtle Man, Bullercruz1, Adrignola and Anonymous: 2
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Game Engine/Game Creating Techniques(GUI)/Creating Pop-Up Menus** *Source:* [https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Game_Engine/Game_Creating_Techniques\(GUI\)/Creating_Pop-Up_Menus?oldid=3101922](https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Game_Engine/Game_Creating_Techniques(GUI)/Creating_Pop-Up_Menus?oldid=3101922) *Contributors:* Turtle Man, Bullercruz1, Muimi~enwikibooks, Adrignola, Avicennasis, Animajosser and Anonymous: 6
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Game Engine/Game Creating Techniques(GUI)/Creating Moving Menus** *Source:* [https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Game_Engine/Game_Creating_Techniques\(GUI\)/Creating_Moving_Menus?oldid=3101923](https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Game_Engine/Game_Creating_Techniques(GUI)/Creating_Moving_Menus?oldid=3101923) *Contributors:* Turtle Man, Bullercruz1, Van der Hoorn, Adrignola, Avicennasis and Animajosser
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Game Engine/Game Creating Techniques(GUI)/The “5-Layer” Button** *Source:* [https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Game_Engine/Game_Creating_Techniques\(GUI\)/The_%225-Layer%22_Button?oldid=3101924](https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Game_Engine/Game_Creating_Techniques(GUI)/The_%225-Layer%22_Button?oldid=3101924) *Contributors:* Turtle Man, Bullercruz1, Adrignola and Animajosser
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Game Engine/Game Creating Techniques(GUI)/Creating Object Outlines** *Source:* [https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Game_Engine/Game_Creating_Techniques\(GUI\)/Creating_Object_Outlines?oldid=3101925](https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Game_Engine/Game_Creating_Techniques(GUI)/Creating_Object_Outlines?oldid=3101925) *Contributors:* Turtle Man, Bullercruz1, Adrignola, Animajosser and Anonymous: 3
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Game Engine/Game Creating Techniques(Python)** *Source:* [https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Game_Engine/Game_Creating_Techniques\(Python\)?oldid=3101949](https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Game_Engine/Game_Creating_Techniques(Python)?oldid=3101949) *Contributors:* Turtle Man, Bullercruz1, Adrignola, Ldo, Animajosser and Anonymous: 1
- **Blender 3D: Noob to Pro/Hacking Blender** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Hacking_Blender?oldid=2646310 *Contributors:* Ldo
- **Blender 3D: Noob to Pro/Intro-GE-Source** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Intro-GE-Source?oldid=3101948 *Contributors:* Venar303, Turtle Man, Adrignola, Avicennasis, Ldo, Animajosser and Anonymous: 1
- **Blender 3D: Noob to Pro/Glossary** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Glossary?oldid=3089142 *Contributors:* ZeroOne, Popski, Snarius~enwikibooks, 1983, Bruyninc~enwikibooks, Rpyle731, Adrignola, Arch dude, Avicennasis, Reyk, Ldo, Stepheng3 and Anonymous: 11
- **Blender 3D: Noob to Pro/FAQ** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/FAQ?oldid=3033067 *Contributors:* Furykef, Alabandit, Jomegat, Jguk, Geoff Plourde, Alex G~enwikibooks, Dolsson5, Ysangkok, Malaz, SoylentGreen, QuiteUnusual, Sigma 7, Manequinho, Adrignola, Rosver~enwikibooks, Illusionist~enwikibooks, LeftClicker, SETH HIKARU, Realdealmagic, JimRester, Matthewrbowker, Msvbdev, Stepheng3, JamesNZ, Mixelpix and Anonymous: 20

- **Blender 3D: Noob to Pro/Tutorial Links List** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Tutorial_Links_List?oldid=3239616 *Contributors:* DavidCary, Argento, Panic2k4, ZeroOne, Popski, Derbeth, EatMyShortz~enwikibooks, Jmartens, 1983, AfroToad, WeirdHat, Super3boy, Joystick Game, GustavTheMushroom, Dbreece, Teddygage, Doj°, Comwizz2~enwikibooks, Vrabecc~enwikibooks, Kok~enwikibooks, Herbythyme, Laleena, Remi0o, IamInnocent, Michaelx153, Recent Runes, Thangalin, Cthames, Chelseafan528, Lechtitseb, Skaruts, Mach1723, RogerWickes, Ok~enwikibooks, Luckofbuck~enwikibooks, NuclearWarfare, QuiteUnusual, Adrignola, Zilch~enwikibooks, PowuaTeam, Reddogzip, WolfprintFX, CrAc, Feithless, Kdjerdon, Hoo man, Ldo, Stepheng3, Petariot, Chandhamama, Volvo2741, User8151, Lolsmurf359, JuethoBot, Houmgaor and Anonymous: 193
- **Blender 3D: Noob to Pro/Hotkeys** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Hotkeys?oldid=2007380 *Contributors:* Popski, Rpyle731, Stepheng3 and Anonymous: 3
- **Blender 3D: Noob to Pro/Output Formats** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Output_Formats?oldid=3262914 *Contributors:* Thenub314, Kloege, Turtle Man, Adrignola, Avicennasis, JimRester, Stepheng3, PokestarFanBot and Anonymous: 1
- **Blender 3D: Noob to Pro/Image Portfolio** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Image_Portfolio?oldid=3226230 *Contributors:* Panic2k4, Siggimund, CommonsDelinker, Pi zero, Turtle Man, Bullercruz1, Adrignola, Electro, Admijr, Pearts, Ldo, JimRester and Stepheng3
- **Blender 3D: Noob to Pro/Blender Glossary** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Blender_Glossary?oldid=3037680 *Contributors:* Nanobug, Bullercruz1, QuiteUnusual, Adrignola, Electro, Stepheng3, Dakantra, Mixelpix and Anonymous: 5
- **Blender 3D: Noob to Pro/Every Material Known to Man** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Every_Material_Known_to_Man?oldid=3243255 *Contributors:* Nanobug, Argento, Spiderworm, Jonon~enwikibooks, Popski, Geri~enwikibooks, Oracleoftruth~enwikibooks, AM088, Soylentgreen, Khan~enwikibooks, Joe 042293, Vault, Super3boy, Jguk, Ktbluear, Davidfg, Mdd4696, Dragonkingme, Chimmay gautam, Xania, JaccoG, Electrice, Bullercruz1, QuiteUnusual, Adrignola, Rct36, Sandsmertz, Dapoulter, Grubham, Ldo, JimRester, Stepheng3, FranklyMyDear..., Atcovi, Win Bigly, Cougar-2022 and Anonymous: 42
- **Blender 3D: Noob to Pro/Sources of free 3D models** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Sources_of_free_3D_models?oldid=3202151 *Contributors:* Jomegat, Turtle Man, Kayau, Billinghurst, Avicennasis, Pearts, Ldo, Techtonik, JFosterKY, Yobi3d, Mavillar stlfinder and Anonymous: 8
- **Blender 3D: Noob to Pro/Blueprint Links List** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Blueprint_Links_List?oldid=3074464 *Contributors:* Argento, Jguk, ThriiDæ, Turtle Man, Adrignola, Pearts, Ldo and Anonymous: 5
- **Blender 3D: Noob to Pro/Materials, Textures, Photos** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Materials%2C_Textures%2C_Photos?oldid=3202035 *Contributors:* Turtle Man, Adrignola, Pearts, Ldo, Vwanweb, Blendergeek and Anonymous: 4
- **Blender 3D: Noob to Pro/Asking for Help** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Asking_for_Help?oldid=2094030 *Contributors:* PhilippWeissenbacher~enwikibooks, Argento, Qrc, Spiderworm, Zoohouse, Crouch, Popski, Derbeth, Lordmyth, Danhash, BimBot, Buckbeak~enwikibooks, Mdd4696, Shellrie, Acsteitz~enwikibooks, Woodsjay~enwikibooks, Tedka, ThomasShepard, Akhram, CarsracBot, Turtle Man, Bullercruz1, Adrignola, JRThro, Stepheng3 and Anonymous: 52
- **Blender 3D: Noob to Pro/Know What You're Doing** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Know_What_You're_Doing?oldid=2094031 *Contributors:* Panic2k4, Turtle Man, Bullercruz1, Sigma 7, Adrignola, Electro, John.delannoy, Colincbn, JRThro, JimRester, Stepheng3 and Anonymous: 21
- **Blender 3D: Noob to Pro/Modeling Realistically** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Modeling_Realistically?oldid=2094035 *Contributors:* Argento, Halley, Jguk, Tybee, Turtle Man, Adrignola, Stepheng3 and Anonymous: 4
- **Blender 3D: Noob to Pro/Modeling tips** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Modeling_tips?oldid=2498204 *Contributors:* SoylentGreen, Turtle Man, Adrignola, Avicennasis, Stepheng3, Cubicsphere and Anonymous: 3
- **Blender 3D: Noob to Pro/Cheat the 3D** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Cheat_the_3D?oldid=2094037 *Contributors:* Nanobug, Turtle Man, Bullercruz1, Adrignola, Mikespedia, Rhettro, Stepheng3 and Anonymous: 16
- **Blender 3D: Noob to Pro/Performance vs. Quality** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Performance_vs._Quality?oldid=2534497 *Contributors:* Turtle Man, Bullercruz1, Velociostrich, Stepheng3 and Anonymous: 6
- **Blender 3D: Noob to Pro/Modeling a Gingerbread Man** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Modeling_a_Gingerbread_Man?oldid=2752479 *Contributors:* Argento, Spiderworm, Matthewr100, Shinjin~enwikibooks, Popski, Astronouth7303, Greggreg, Yosefm~enwikibooks, Knight13, AM088, Danielsimlind, Darkonc, Jomegat, SchmS, NathanStocks, Milky~enwikibooks, Majko02, Hagindaz, SatanClaus~enwikibooks, Orbisonitrum, N comme Nul, Tannersrf, Moreejt, Dolsson5, CommonsDelinker, TBOL3, Mike.lifeguard, Swap i, Brylie, SergeantOreo, Firecentaur~enwikibooks, Tyaedalis, Vaxine19, Turtle Man, Bullercruz1, Rpyle731, Adrignola, Thorack~enwikibooks, Camp, Mrwilley, Khono, Bonapon, 80063r, DDD3x, Worm 19, Stepheng3, Tadmuck and Anonymous: 86
- **Blender 3D: Noob to Pro/Modeling a simple space-ship** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Modeling_a_simple_space-ship?oldid=2290192 *Contributors:* Turtle Man, Adrignola, J36miles, GreenBlobWiki and Jfmantis
- **Blender 3D: Noob to Pro/Create an animated GIF wallpaper (Blender/GIMP)** *Source:* [https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Create_an_animated_GIF_wallpaper_\(Blender/GIMP\)?oldid=2545904](https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Create_an_animated_GIF_wallpaper_(Blender/GIMP)?oldid=2545904) *Contributors:* Vesseshhebar
- **Blender 3D: Noob to Pro/Part 1 - Preparing the Scene** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Part_1_-_Preparing_the_Scene?oldid=3263071 *Contributors:* Vesseshhebar and PokestarFanBot
- **Blender 3D: Noob to Pro/Creating Weapons based on 2D Images** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Creating_Weapons_based_on_2D_Images?oldid=2654194 *Contributors:* Swift, Mike.lifeguard, Chelseafan528, Morerunes, Embri, Root~enwikibooks, SoylentGreen, QuiteUnusual, Spellmaker, Null Point, Totalwriter0331, Ldo and Anonymous: 5
- **Blender 3D: Noob to Pro/Making Your Creation Smoother** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Making_Your_Creation_Smoother?oldid=2561219 *Contributors:* Xemkis12, Bullercruz1, Gairlochan~enwikibooks, Adrignola, Arch dude, Null Point, Avicennasis, Pearts, Ldo, Neverbirth and Anonymous: 14

- **Blender 3D: Noob to Pro/Match Moving** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Match_Moving?oldid=2290191 *Contributors:* Chuckhoffmann, Dan-aka-jack, Mike.lifeguard, SergeantOreo, Turtle Man, Lordelph~enwikibooks, Adrignola and Anonymous: 8
- **Blender 3D: Noob to Pro/Motion Tracking with Icarus** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Motion_Tracking_with_Icarus?oldid=3227971 *Contributors:* Turtle Man, Tom soderlund~enwikibooks, Adrignola, Pixelstuff and Anonymous: 6
- **Blender 3D: Noob to Pro/Create a Clayman** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Create_a_Clayman?oldid=2264405 *Contributors:* Iamunknow, Darklama, SBJohnny, Aschoeke, Swift, Bootaleg, Tannersf, Herbythyme, Reece, Cptlongshlong, Turtle Man, Bullercruz1, QuiteUnusual, Nick.anderegg, Avicennasis and Anonymous: 5
- **Blender 3D: Noob to Pro/Organic Modeling** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Organic_Modeling?oldid=2094105 *Contributors:* Panic2k4, Pyro guy, Jguk, CommonsDelinker, Venar303, Turtle Man, Tdunn7, Adrignola, Admijr and Anonymous: 2
- **Blender 3D: Noob to Pro/Understanding the Fluid Simulator** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Understanding_the_Fluid_Simulator?oldid=3124377 *Contributors:* Recent Runes, Turtle Man, Kayau, Adrignola, Orfo~enwikibooks, Woodstocket, Avicennasis, HakanIST and Anonymous: 9
- **Blender 3D: Noob to Pro/Creating a jewel in Blender** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Creating_a_jewel_in_Blender?oldid=2094108 *Contributors:* Webaware, CommonsDelinker, Mike.lifeguard, Wildern, Turtle Man, Jensenns, Adrignola, Orfo~enwikibooks, Sendoshin, GeForce3, Sapalskimichal and Anonymous: 4
- **Blender 3D: Noob to Pro/Modeling a picture** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Modeling_a_picture?oldid=2094109 *Contributors:* Faux~enwikibooks, AM088, Withinfocus, SchmS, JacKDUDrEn~enwikibooks, Jguk, Reverzantative, Az1568, Rm5248, Bmarnz, SoylentGreen, Turtle Man, Bullercruz1, Adrignola, Avicennasis and Anonymous: 25
- **Blender 3D: Noob to Pro/Modeling with the Spin Tool** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Modeling_with_the_Spin_Tool?oldid=2094110 *Contributors:* Turtle Man, Adrignola, Rosver~enwikibooks and Anonymous: 2
- **Blender 3D: Noob to Pro/Spin Tool Introduction** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Spin_Tool_Introduction?oldid=3290777 *Contributors:* Panic2k4, Turtle Man, Sigma 7, Adrignola, Rosver~enwikibooks, Avicennasis, Reyk and Anonymous: 6
- **Blender 3D: Noob to Pro/Illustrative Example: Model a Wine Glass** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Illustrative_Example%3A_Model_a_Wine_Glass?oldid=2163102 *Contributors:* Thenub314, Adrignola, Rosver~enwikibooks and Anonymous: 1
- **Blender 3D: Noob to Pro/Creating Ogg-Theora movies using Blender** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Creating_Ogg-Theora_movies_using_Blender?oldid=2163103 *Contributors:* Inductiveload, Rwst~enwikibooks, Adrignola, Avicennasis and Anonymous: 1
- **Blender 3D: Noob to Pro/Creating animated GIFs using Blender and Gimp** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Creating_animated_GIFs_using_Blender_and_Gimp?oldid=3217327 *Contributors:* Socratesone, Darklama, Aidan.Sullivan, DrDixie, Chelseafan528, Soeb, Adrignola, Quibik, Doakey3 and Anonymous: 3
- **Blender 3D: Noob to Pro/3D Tiling Backgrounds For The Web** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/3D_Tiling_Backgrounds_For_The_Web?oldid=2163110 *Contributors:* Quantum Anomaly, Red4tribe, Adrignola, Avicennasis and Anonymous: 3
- **Blender 3D: Noob to Pro/Cool Things** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Cool_Things?oldid=3208403 *Contributors:* Jomegat, Iluvblender, Milky~enwikibooks, Jguk, Mstram, Jose da vinci~enwikibooks, Fangiotophia, Reece, Tybee, Adrignola, Avicennasis, Reyk, Ldo and Anonymous: 35
- **Blender 3D: Noob to Pro/Troubleshooting** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Troubleshooting?oldid=2197554 *Contributors:* Jguk, Thenub314, Dallas1278, Adrignola, Avicennasis and Anonymous: 7
- **Blender 3D: Noob to Pro/Creating Blender Libraries** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Creating_Blender_Libraries?oldid=3182083 *Contributors:* Argento, Halley, Jguk, Adrignola, Avicennasis, JamesNZ and Anonymous: 7
- **Blender 3D: Noob to Pro/Add some depth with stereo** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Add_some_depth_with_stereo?oldid=2753804 *Contributors:* Jguk, Adrignola, Ftiercel, Hadphild, Leaderboard, Suspender guy and Anonymous: 11
- **Blender 3D: Noob to Pro/Fluffy Material** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Fluffy_Material?oldid=2059784 *Contributors:* Jonon~enwikibooks, Trident~enwikibooks, Jguk, Radialronnie, Adrignola, Avicennasis and Anonymous: 1
- **Blender 3D: Noob to Pro/Human Body** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Human_Body?oldid=3239617 *Contributors:* Argento, Jguk, Adrignola, Rgriffogoes, Frank H, Avicennasis, Ldo, Othmanskn2, Houmgaor and Anonymous: 3
- **Blender 3D: Noob to Pro/Rendering Information** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Rendering_Information?oldid=1616990 *Contributors:* Argento, Jguk and Adrignola
- **Blender 3D: Noob to Pro/Using Blender Libraries** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Using_Blender_Libraries?oldid=1616668 *Contributors:* Argento, Darklama, Adrignola and Anonymous: 2
- **Blender 3D: Noob to Pro/Beginning Modeling Final Project** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Beginning_Modeling_Final_Project?oldid=1616479 *Contributors:* Spiderworm, Sheridan, Popski, Jguk, Mattb112885, Rm5248, MSK61, Icky-wiki, Yjvyas, Bullercruz1, Adrignola and Anonymous: 12
- **Blender 3D: Noob to Pro/Using Inkscape to make advanced Bezier curves** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Using_Inkscape_to_make_advanced_Bezier_curves?oldid=3005471 *Contributors:* Thenub314, QuiteUnusual, Adrignola, Tjb0607, Avicennasis, Perhelion, Slashme, Addihockey10 (automated) and Anonymous: 4
- **Blender 3D: Noob to Pro/Advanced Tutorials/Advanced Modeling/Light Mapping** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Advanced_Tutorials/Advanced_Modeling/Light_Mapping?oldid=2093994 *Contributors:* Turtle Man, Bullercruz1 and Adrignola

- **Blender 3D: Noob to Pro/Platonic Solids** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Platonic_Solids?oldid=3071845 *Contributors:* Reyk, Ldo, Nkansahrexford and Anonymous: 4
- **Blender 3D: Noob to Pro/Polygonal Modeling** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Polygonal_Modeling?oldid=3044698 *Contributors:* Thenub314, Adrignola, Rosver~enwikibooks and Shiroi Kyuketsuki
- **Blender 3D: Noob to Pro/Box Modeling** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Box_Modeling?oldid=3037417 *Contributors:* Jomegat, Jguk, Thenub314, Recent Runes, Adrignola, Rosver~enwikibooks, Avicennasis, Ldo, HakanIST and Anonymous: 9
- **Blender 3D: Noob to Pro/Illustrative example: Model a Chair (Swan Chair)** *Source:* [https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Illustrative_example%3A_Model_a_Chair_\(Swan_Chair\)?oldid=1617775](https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Illustrative_example%3A_Model_a_Chair_(Swan_Chair)?oldid=1617775) *Contributors:* Thenub314, Adrignola, Rosver~enwikibooks and Anonymous: 1
- **Blender 3D: Noob to Pro/Model a Chair-Preparations** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Model_a_Chair-Preparations?oldid=1617765 *Contributors:* Panic2k4, Adrignola and Rosver~enwikibooks
- **Blender 3D: Noob to Pro/Model a Chair-The Seat** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Model_a_Chair-The_Seat?oldid=2059879 *Contributors:* Adrignola, Rosver~enwikibooks, Avicennasis and Anonymous: 2
- **Blender 3D: Noob to Pro/Model a Chair-The Feet** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Model_a_Chair-The_Feet?oldid=1949012 *Contributors:* PartierSP, Adrignola, Rosver~enwikibooks and Anonymous: 1
- **Blender 3D: Noob to Pro/Illustrative Example: Modeling a Simple Human Character** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Illustrative_Example%3A_Modeling_a_Simple_Human_Character?oldid=1694690 *Contributors:* Adrignola, Rosver~enwikibooks and Anonymous: 2
- **Blender 3D: Noob to Pro/Modeling a Human Character - Preparations** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Modeling_a_Human_Character_-_Preparations?oldid=1694692 *Contributors:* Adrignola and Rosver~enwikibooks
- **Blender 3D: Noob to Pro/Modeling a Human Character - Modeling** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Modeling_a_Human_Character_-_Modeling?oldid=2769048 *Contributors:* Adrignola, Rosver~enwikibooks, Reyk, Glaisher and Anonymous: 4
- **Blender 3D: Noob to Pro/Polygon by Polygon modeling** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Polygon_by_Polygon_modeling?oldid=2294024 *Contributors:* Thenub314, Adrignola, Rosver~enwikibooks, Avicennasis and Anonymous: 5
- **Blender 3D: Noob to Pro/Animation Notes and FAQ** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Animation_Notes_and_FAQ?oldid=2234131 *Contributors:* Thenub314, Mike.lifeguard, Theteach, Dallas1278, Adrignola, Ironicaby, Left-Clicker, Avicennasis and Anonymous: 3
- **Blender 3D: Noob to Pro/Customization** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Customization?oldid=2094042 *Contributors:* Turtle Man and Stepheng3
- **Blender 3D: Noob to Pro/Mist - Make Objects Opaque** *Source:* https://en.wikibooks.org/wiki/Blender_3D%3A_Noob_to_Pro/Mist_-_Make_Objects_Opaque?oldid=2116053 *Contributors:* CommonsDelinker, Turtle Man, Adrignola, J36miles and Anonymous: 4

6.2 Images

- **File:00%.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d6/00%25.svg> *License:* Public domain *Contributors:* Based on the XML code of [Image:25%.svg](#) *Original artist:* Siebrand
- **File:00_percents.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/60/00_percent.svg *License:* CC0 *Contributors:* File:00%.svg *Original artist:* Siebrand
- **File:100_percent.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/24/100_percent.svg *License:* CC0 *Contributors:* File:100%.svg *Original artist:* Siebrand
- **File:100_percents.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/24/100_percent.svg *License:* CC0 *Contributors:* File:100%.svg *Original artist:* Siebrand
- **File:25_percent.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/ce/25_percent.svg *License:* CC0 *Contributors:* File:25%.svg *Original artist:* Ftiercel
- **File:25_percents.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/ce/25_percent.svg *License:* CC0 *Contributors:* File:25%.svg *Original artist:* Ftiercel
- **File:3_norms_material.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/19/3_norms_material.jpg *License:* GPL *Contributors:* Multiple/Spliced Blender 3D snapshots *Original artist:* Jeremy B.
- **File:3d_Modelled_st...l_goblets2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/ba/3d_Modelled_st...l_goblets2.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Taabishm2
- **File:3dbg-boundaries.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/65/3dbg-boundaries.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:3dbg-copypaste.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/3a/3dbg-copypaste.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:3dbg-deletefaces.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/02/3dbg-deletefaces.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks

- **File:3dbg-eraseseam.gif** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a3/3dbg-eraseseam.gif> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:3dbg-extrude&scale.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/0d/3dbg-extrude%26scale.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:3dbg-finaltile.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/3e/3dbg-finaltile.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Quantum Anomaly at en.wikibooks. Later version(s) were uploaded by Adrignola at en.wikibooks.
- **File:3dbg-guides.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/66/3dbg-guides.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:3dbg-orthocamera.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/cf/3dbg-orthocamera.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Quantum Anomaly at en.wikibooks
- **File:3dbg-persp-ortho.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/3a/3dbg-persp-ortho.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:3dbg-planeblack.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/99/3dbg-planeblack.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:3dbg-planescale.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f1/3dbg-planescale.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:3dbg-populated.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/6c/3dbg-populated.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Quantum Anomaly at en.wikibooks
- **File:3dbg-scale&dup.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/7d/3dbg-scale%26dup.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Quantum Anomaly at en.wikibooks
- **File:3dbg-tilepoints.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/4e/3dbg-tilepoints.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:3dbg-zmove.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a1/3dbg-zmove.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:3rd_face.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/eb/3rd_face.jpg *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Badalia at English Wikibooks
- **File:50_percents.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/62/50_percents.svg *License:* CC0 *Contributors:* File: 50%.svg *Original artist:* Ftiercel
- **File:5cutslide.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/37/5cutslide.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* JonnyJack
- **File:75_percent.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/62/75_percent.svg *License:* CC0 *Contributors:* File: 75%.svg *Original artist:* Ftiercel
- **File:75_percents.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/62/75_percent.svg *License:* CC0 *Contributors:* File: 75%.svg *Original artist:* Ftiercel
- **File:9_posiciones.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ef/9_posiciones.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Reptiles extintos at English Wikibooks
- **File:A01_DEFAULTS.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/05/A01_DEFAULTS.png *License:* CC BY-SA 3.0 *Contributors:* Screenshot of the GPL program Blender showing various types of lights *Original artist:* MercedMike
- **File:A02_combined_energy.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6b/A02_combined_energy.png *License:* CC BY-SA 3.0 *Contributors:* Screenshot of the GPL program Blender showing lighting variations *Original artist:* MercedMike
- **File:A04_combined_height.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/84/A04_combined_height.png *License:* CC BY-SA 3.0 *Contributors:* Screenshot of the GPL program Blender showing light variations *Original artist:* MercedMike
- **File:Action-1.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/4c/Action-1.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Halley at en.wikibooks
- **File:Action-2.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2c/Action-2.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Halley at en.wikibooks
- **File:Adding_new_faces-1.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a8/Adding_new_faces-1.png *License:* Public domain *Contributors:* Desktop screen capture, my PC at home *Original artist:* Paul James Chatman
- **File:AfterSpinning.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9a/AfterSpinning.png> *License:* Public domain *Contributors:* Own work *Original artist:* blender Foundation

- **File:After_add_new_F5.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/52/After_add_new_F5.jpg *License:* GPL *Contributors:* Own work *Original artist:* John Whelan
- **File:Allcutslide.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/4f/Allcutslide.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* JonnyJack
- **File:Alt-m-five-spots.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/26/Alt-m-five-spots.png> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Wynnmc at en.wikibooks
- **File:Anim_tab.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e6/Anim_tab.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Soeb at English Wikibooks
- **File:AnimatingLattice01.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/44/AnimatingLattice01.png> *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:AnimatingLattice03.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d6/AnimatingLattice03.png> *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:AnimatingLattice04.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/de/AnimatingLattice04.png> *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:AnimatingLattice05.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/bd/AnimatingLattice05.png> *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:AnimatingLattice06.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1e/AnimatingLattice06.png> *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:AnimatingLattice07.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9d/AnimatingLattice07.png> *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:AnimatingLattice08.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/33/AnimatingLattice08.png> *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:AnimatingLattice09.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a5/AnimatingLattice09.png> *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:AnimatingLattice10.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1a/AnimatingLattice10.png> *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:Armature_3D_Header.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/1a/Armature_3D_Header.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Gabio at English Wikibooks
- **File:Armature_button_obj.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b8/Armature_button_obj.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Gabio at English Wikibooks
- **File:Axis_drawing.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/30/Axis_drawing.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:BNTP_img1.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6c/BNTP_img1.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Radialronnie at English Wikibooks
- **File:BNTP_img10.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0d/BNTP_img10.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Radialronnie at English Wikibooks
- **File:BNTP_img11.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/4b/BNTP_img11.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Radialronnie at English Wikibooks
- **File:BNTP_img13.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0f/BNTP_img13.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Radialronnie at English Wikibooks
- **File:BNTP_img2.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a3/BNTP_img2.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Radialronnie at English Wikibooks
- **File:BNTP_img3.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/60/BNTP_img3.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Radialronnie at English Wikibooks
- **File:Bake-basic-fluid-tutorial-blender2.76.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/50/Bake-basic-fluid-tutorial-blender2.76.jpg> *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Bake_And_Save.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/74/Bake_And_Save.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Quantum Anomaly at English Wikibooks
- **File:Ball3_1_2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/51/Ball3_1_2.jpg *License:* CC BY-SA 3.0 *Contributors:* http://en.wikipedia.org/wiki/Image:HDR_example_-_exposure.jpeg *Original artist:* Wayne Harrison
- **File:Basic-scene.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/40/Basic-scene.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Originally from en.wikibooks; description page is/was here. *Original artist:* Original uploader was Dan13 at en.wikibooks
- **File:Bbones.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/af/Bbones.jpg> *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Gabio at en.wikibooks
- **File:Bcup1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b3/Bcup1.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was AverieZen at English Wikibooks

- **File:Beocup10.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/cf/Beocup10.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Beocup11.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/6e/Beocup11.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Beocup12.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b6/Beocup12.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Beocup14.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a4/Beocup14.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Beocup15.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/4d/Beocup15.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Beocup3.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ae/Beocup3.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Beocup4.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9a/Beocup4.png> *License:* GPL *Contributors:* Transferred from en.wikibooks *Original artist:* Original uploader was AverieZen at en.wikibooks
- **File:Beocup5.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f5/Beocup5.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Beocup6.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a3/Beocup6.png> *License:* GPL *Contributors:* Transferred from en.wikibooks *Original artist:* Original uploader was AverieZen at en.wikibooks
- **File:Beocup9.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b5/Beocup9.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:BeforeSpining.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a8/BeforeSpining.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:Beizersharpturn0003.gif** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/10/Beizersharpturn0003.png> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Jawboot at English Wikibooks
- **File:Bevel-done.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/44/Bevel-done.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Wynnmc at English Wikibooks
- **File:Bezier_Face_Bevel.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ee/Bezier_Face_Bevel.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Lazy-lump at English Wikibooks
- **File:Bezier_Face_Bevel_4.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/02/Bezier_Face_Bevel_4.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Lazy-lump at English Wikibooks
- **File:Bezier_circle1.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/58/Bezier_circle1.png *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Kieran at English Wikibooks
- **File:Bezier_circle2.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a6/Bezier_circle2.png *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Kieran at English Wikibooks
- **File:Bezier_circle3.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b9/Bezier_circle3.png *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Kieran at English Wikibooks
- **File:Bezier_circle4.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/7c/Bezier_circle4.png *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Kieran at English Wikibooks
- **File:Bezier_circle_10b.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f9/Bezier_circle_10b.png *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Kieran at English Wikibooks
- **File:Bezier_circle_5.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a7/Bezier_circle_5.png *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Kieran at English Wikibooks
- **File:Bezier_circle_6.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/11/Bezier_circle_6.png *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Kieran at en.wikibooks
- **File:Bezier_circle_8.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/23/Bezier_circle_8.png *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Kieran at en.wikibooks
- **File:Bezier_circle_9b.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/2d/Bezier_circle_9b.png *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Kieran at en.wikibooks

- **File:Bezler_Geometry.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/53/Bezler_Geometry.png *License:* GPL *Contributors:* Bobhaspantz (https://commons.wikimedia.org/wiki/User_talk:Bobhaspantz)' title='User talk:Bobhaspantz'>talk) 19:02, 20 March 2017 (UTC) *Original artist:* ?
- **File:Bird_149_mist_distance-1_0400.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/70/Bird_149_mist_distance-1_0400.jpg *License:* GPL *Contributors:* Blender *Original artist:* ?
- **File:Bird_149_mist_distance-2_0400.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/85/Bird_149_mist_distance-2_0400.jpg *License:* GPL *Contributors:* Blender *Original artist:* ?
- **File:Bird_149_mist_distance-3_0400.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d6/Bird_149_mist_distance-3_0400.jpg *License:* GPL *Contributors:* Blender *Original artist:* ?
- **File:Bird_149_mist_distance-4_0400.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/2f/Bird_149_mist_distance-4_0400.jpg *License:* GPL *Contributors:* Blender *Original artist:* ?
- **File:Bird_Blender%27sMist_result-1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/cc/Bird_Blender%27sMist_result-1.jpg *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Kome111
- **File:Bl1.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/cb/Bl1.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl10.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a7/Bl10.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl11.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/58/Bl11.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl12.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/4b/Bl12.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl13.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/39/Bl13.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl14.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/19/Bl14.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl15.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/fb/Bl15.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl16.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d0/Bl16.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Sumoncil at en.wikibooks. Later version(s) were uploaded by Adrignola at en.wikibooks.
- **File:Bl18.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/10/Bl18.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl19.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/45/Bl19.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl2.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/bf/Bl2.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Sumoncil at en.wikibooks. Later version(s) were uploaded by Adrignola at en.wikibooks.
- **File:Bl20.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5d/Bl20.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl21.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f0/Bl21.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Sumoncil at en.wikibooks. Later version(s) were uploaded by Adrignola at en.wikibooks.
- **File:Bl22.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/65/Bl22.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl24.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c1/Bl24.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl25.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/7c/Bl25.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl26.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a0/Bl26.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks

- **File:Bl27.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9d/Bl27.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl28.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/67/Bl28.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl3.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/cd/Bl3.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl30.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/04/Bl30.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl31.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/21/Bl31.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl32.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2a/Bl32.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl33.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/ec/Bl33.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Sumoncil at en.wikibooks. Later version(s) were uploaded by Adrignola at en.wikibooks.
- **File:Bl34.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/63/Bl34.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Sumoncil at en.wikibooks. Later version(s) were uploaded by Adrignola at en.wikibooks.
- **File:Bl35.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/8e/Bl35.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl36.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/35/Bl36.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl37.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2f/Bl37.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl39.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a9/Bl39.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl4.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5d/Bl4.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl5.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ae/Bl5.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl6.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9b/Bl6.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl7.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/12/Bl7.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl8.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/26/Bl8.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Bl9.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a4/Bl9.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Sumoncil at English Wikibooks
- **File:Blender'{}s_Node_Editor_anaglyph_diagram.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/38/Blender%27s_Node_Editor_anaglyph_diagram.png *License:* GFDL *Contributors:* Own work *Original artist:* Fabrice TIERCELIN
- **File:Blender-2.57_headers.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/11/Blender-2.57_headers.jpg *License:* CC0 *Contributors:* Own work *Original artist:* JamesNZ (JamesNZ (talk))
- **File:Blender-2.57_viewport_outline.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a7/Blender-2.57_viewport_outline.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_cube_edit_mode.PNG** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/68/Blender-2.5_cube_edit_mode.PNG *License:* CC0 *Contributors:* Own work *Original artist:* JamesNZ
- **File:Blender-2.5_cube_object_mode.PNG** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5e/Blender-2.5_cube_object_mode.PNG *License:* CC0 *Contributors:* Own work *Original artist:* JamesNZ

- **File:Blender-2.5_flying_through_canyon_applied_displace.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/3c/Blender-2.5_flying_through_canyon_applied_displace.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_flying_through_canyon_finished.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0b/Blender-2.5_flying_through_canyon_finished.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_flying_through_canyon_finished_model.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/76/Blender-2.5_flying_through_canyon_finished_model.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_flying_through_canyon_subdivided_plane.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e0/Blender-2.5_flying_through_canyon_subdivided_plane.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_lamp_radio_button.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f3/Blender-2.5_lamp_radio_button.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_layers.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/75/Blender-2.5_layers.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_materials.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/33/Blender-2.5_materials.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_mode_menu.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5f/Blender-2.5_mode_menu.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_occlude_geometry.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/63/Blender-2.5_occlude_geometry.png *License:* CC0 *Contributors:* Own work *Original artist:* JamesNZ
- **File:Blender-2.5_outliner.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/92/Blender-2.5_outliner.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_parenting_objects.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d9/Blender-2.5_parenting_objects.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_perfect_view_quad.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/74/Blender-2.5_perfect_view_quad.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_polycount.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/67/Blender-2.5_polycount.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_quickie_model_render.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/be/Blender-2.5_quickie_model_render.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_quickie_model_stage2.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fa/Blender-2.5_quickie_model_stage2.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_quickie_model_stage3.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ef/Blender-2.5_quickie_model_stage3.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_quickie_model_stage4.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/bb/Blender-2.5_quickie_model_stage4.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_quickie_model_stage5.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/55/Blender-2.5_quickie_model_stage5.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_quickie_model_stage6.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/2f/Blender-2.5_quickie_model_stage6.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_quikiemodel_advanced_stage1.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a8/Blender-2.5_quikiemodel_advanced_stage1.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_renaming_hat.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fb/Blender-2.5_renaming_hat.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_saving_quickie_model.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/ab/Blender-2.5_saving_quickie_model.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_simple_hat.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/af/Blender-2.5_simple_hat.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_simple_person_arms_up.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8b/Blender-2.5_simple_person_arms_up.png *License:* CC0 *Contributors:* Own work *Original artist:* JamesNZ
- **File:Blender-2.5_simple_person_delete_arm.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0f/Blender-2.5_simple_person_delete_arm.png *License:* CC0 *Contributors:* Own work *Original artist:* JamesNZ
- **File:Blender-2.5_simple_person_detailed.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0c/Blender-2.5_simple_person_detailed.png *License:* CC0 *Contributors:* Own work *Original artist:* JamesNZ
- **File:Blender-2.5_simple_person_detailed2.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a4/Blender-2.5_simple_person_detailed2.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_simple_person_detailed2_feet.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b6/Blender-2.5_simple_person_detailed2_feet.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_simple_person_detailed2_torso_scale.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d3/Blender-2.5_simple_person_detailed2_torso_scale.png *License:* CC0 *Contributors:* Own work *Original artist:* JamesNZ
- **File:Blender-2.5_simple_person_geometric_center.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/57/Blender-2.5_simple_person_geometric_center.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_simple_person_hat.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/43/Blender-2.5_simple_person_hat.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation

- **File:Blender-2.5_simple_person_snapping.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/57/Blender-2.5_simple_person_snapping.png *License:* CC0 *Contributors:* Self-taken. *Original artist:* Blender Foundation
- **File:Blender-2.5_simple_person_step2.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5b/Blender-2.5_simple_person_step2.png *License:* CC0 *Contributors:* Own work *Original artist:* JamesNZ
- **File:Blender-2.5_simple_person_step3.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/ce/Blender-2.5_simple_person_step3.png *License:* CC0 *Contributors:* Own work *Original artist:* JamesNZ
- **File:Blender-2.5_simple_person_step4.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/be/Blender-2.5_simple_person_step4.png *License:* CC0 *Contributors:* Own work *Original artist:* JamesNZ
- **File:Blender-2.5_simple_person_step5.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/62/Blender-2.5_simple_person_step5.png *License:* CC0 *Contributors:* Own work *Original artist:* JamesNZ
- **File:Blender-2.5_snap_menu.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/93/Blender-2.5_snap_menu.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_user_preferences.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8a/Blender-2.5_user_preferences.png *License:* CC0 *Contributors:* Self-taken *Original artist:* ?
- **File:Blender-2.5_user_preferences_editing.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c5/Blender-2.5_user_preferences_editing.png *License:* GPL *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_user_preferences_files.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/bc/Blender-2.5_user_preferences_files.png *License:* CC0 *Contributors:* Self-taken *Original artist:* ?
- **File:Blender-2.5_user_preferences_input.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a9/Blender-2.5_user_preferences_input.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.5_view_menu.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d9/Blender-2.5_view_menu.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.62-Adding-an-Armature-1.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e0/Blender-2.62-Adding-an-Armature-1.jpg> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Nkansah Rexford
- **File:Blender-2.62-Adding-an-Armature-2.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/be/Blender-2.62-Adding-an-Armature-2.jpg> *License:* CC BY-SA 3.0 *Contributors:* My own work *Original artist:* Nkansah Rexford
- **File:Blender-2.6_change_light_type.PNG** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b6/Blender-2.6_change_light_type.PNG *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Blender-2.76_flying_through_canyon_displaced_plane.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5a/Blender-2.76_flying_through_canyon_displaced_plane.jpg *License:* GPL *Contributors:* Own work *Original artist:* Animajosser
- **File:Blender-2dlogo-polish-final.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/24/Blender-2dlogo-polish-final.png> *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* Mrfelis
- **File:Blender-2dlogo-polish-find-defects.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/54/Blender-2dlogo-polish-find-defects.png> *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* Mrfelis
- **File:Blender-2dlogo-polish-fix-defects.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/22/Blender-2dlogo-polish-fix-defects.png> *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* Mrfelis
- **File:Blender-2dlogo-polish-position-2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2a/Blender-2dlogo-polish-position-2.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Mrfelis at English Wikibooks
- **File:Blender-2dlogo-polish-position-control-points.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f2/Blender-2dlogo-polish-position-control-points.png> *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* Mrfelis
- **File:Blender-2dlogo-rough-trace-final.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/09/Blender-2dlogo-rough-trace-final.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Mrfelis at English Wikibooks
- **File:Blender-2dlogo-rough-trace-points.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e2/Blender-2dlogo-rough-trace-points.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Mrfelis at English Wikibooks
- **File:Blender-3d-cursor.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e1/Blender-3d-cursor.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-DefaultSceneRender.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/98/Blender-DefaultSceneRender.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Nuclearpidgeon at English Wikibooks
- **File:Blender-LimitSelectionToVisible.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/3b/Blender-LimitSelectionToVisible.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* PvtKing
- **File:Blender-Noob-to-Pro_Proportional_Menu.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/85/Blender-Noob-to-Pro_Proportional_Menu.png *License:* Public domain *Contributors:* Transferred from en.wikibooks *Original artist:* Original uploader was Lirtferk at en.wikibooks
- **File:Blender-ObjectID-IndexPass-Settings_BlenderMistTutorial.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0b/Blender-ObjectID-IndexPass-Settings_BlenderMistTutorial.jpg *License:* GPL *Contributors:* Blender *Original artist:* ?
- **File:Blender-OriginTo3dCursor.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f2/Blender-OriginTo3dCursor.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* PvtKing

- **File:Blender-add-menu.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/dc/Blender-add-menu.png> *License:* GPL
Contributors: Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-appli2biped-workstation.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/3e/Blender-appli2biped-workstation.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-auto-perspective.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2e/Blender-auto-perspective.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-background-dialog.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/44/Blender-background-dialog.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Mrfelis at English Wikibooks
- **File:Blender-bezier-curve-initial.PNG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/08/Blender-bezier-curve-initial.PNG> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Mrfelis at English Wikibooks
- **File:Blender-bezier-curve-sample.PNG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/3f/Blender-bezier-curve-sample.PNG> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Mrfelis at English Wikibooks
- **File:Blender-browse-button.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/25/Blender-browse-button.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-browser-show-more.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/17/Blender-browser-show-more.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-curve-circle.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/61/Blender-curve-circle.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Guillaume22
- **File:Blender-curve-path.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/36/Blender-curve-path.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Guillaume22
- **File:Blender-curve-tools-convert.PNG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a7/Blender-curve-tools-convert.PNG> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Mrfelis at English Wikibooks
- **File:Blender-default-scene.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/cf/Blender-default-scene.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-edit-method-panel.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/db/Blender-edit-method-panel.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-edit-select-modes.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b0/Blender-edit-select-modes.png> *License:* Public domain *Contributors:* Guillaume22 *Original artist:* ?
- **File:Blender-export.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/06/Blender-export.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-header.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/7d/Blender-header.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-landscape_with_height_maps(1).png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c7/Blender-landscape_with_height_maps%281%29.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was Aidan.Sullivan at English Wikibooks
- **File:Blender-landscape_with_height_maps.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/2b/Blender-landscape_with_height_maps.png *License:* GPL *Contributors:* Transferred from en.wikibooks *Original artist:* Original uploader was Aidan.Sullivan at en.wikibooks
- **File:Blender-light-spot.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b1/Blender-light-spot.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Guillaume22
- **File:Blender-mesh-cone.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/10/Blender-mesh-cone.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Guillaume22
- **File:Blender-mesh-cylinder.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/7d/Blender-mesh-cylinder.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Guillaume22
- **File:Blender-mesh-panel.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/7e/Blender-mesh-panel.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-mesh-suzanne.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5e/Blender-mesh-suzanne.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Guillaume22
- **File:Blender-mesh-torus.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c3/Blender-mesh-torus.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Guillaume22
- **File:Blender-meta-ellipsoid.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f4/Blender-meta-ellipsoid.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Guillaume22
- **File:Blender-meta-plane.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d5/Blender-meta-plane.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Guillaume22
- **File:Blender-oops-shematic.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2f/Blender-oops-shematic.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-outliner-cube-group.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ad/Blender-outliner-cube-group.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation

- **File:Blender-parent-outliner.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/58/Blender-parent-outliner.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-refresh-pyton-menus.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/db/Blender-refresh-pyton-menus.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-seawater.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/21/Blender-seawater.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Dobz116
- **File:Blender-show-python.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c0/Blender-show-python.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-show-video.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/ec/Blender-show-video.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-system-openGL-panel.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/59/Blender-system-openGL-panel.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-transform-properties.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a8/Blender-transform-properties.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender-windows-header.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/67/Blender-windows-header.png> *License:* GPL *Contributors:* Guillaume22 *Original artist:* Blender Foundation
- **File:Blender2.75a_Pyramid_background_image.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/35/Blender2.75a_Pyramid_background_image.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Blender2.75a_make_faces_background_image.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/04/Blender2.75a_make_faces_background_image.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Blender2.75a_make_faces_made_background_image.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/32/Blender2.75a_make_faces_made_background_image.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Blender2.75a_make_vertices_background_image.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/3a/Blender2.75a_make_vertices_background_image.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Blender2.76_Video_sequencer_1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/3e/Blender2.76_Video_sequencer_1.jpg *License:* GPL *Contributors:* Own work *Original artist:* Animajosser
- **File:Blender2.76_Video_sequencer_2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e9/Blender2.76_Video_sequencer_2.jpg *License:* GPL *Contributors:* Own work *Original artist:* Animajosser
- **File:Blender2.76_Video_sequencer_3.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/35/Blender2.76_Video_sequencer_3.jpg *License:* GPL *Contributors:* Own work *Original artist:* Animajosser
- **File:Blender241.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c1/Blender241.png> *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* Alex Grace
- **File:Blender254WindowWidget.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/41/Blender254WindowWidget.png> *License:* CC0 *Contributors:* Blender 2.54 *Original artist:* Lawrence D'Oliveiro
- **File:Blender255MaterialContextButton.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/fd/Blender255MaterialContextButton.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* ?
- **File:Blender255ObjectConstraintsContextButton.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9a/Blender255ObjectConstraintsContextButton.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* ?
- **File:Blender255ObjectContextButton.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/82/Blender255ObjectContextButton.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* ?
- **File:Blender255ObjectModifiersContextButton.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5c/Blender255ObjectModifiersContextButton.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* ?
- **File:Blender255ParticlesContextButton.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a6/Blender255ParticlesContextButton.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* ?
- **File:Blender255PhysicsContextButton.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/75/Blender255PhysicsContextButton.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* ?
- **File:Blender255RenderContextButton.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/64/Blender255RenderContextButton.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* ?
- **File:Blender255SceneContextButton.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/79/Blender255SceneContextButton.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* ?
- **File:Blender255TextureContextButton.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d0/Blender255TextureContextButton.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* ?
- **File:Blender255WorkspaceMenu.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/0b/Blender255WorkspaceMenu.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* Lawrence D'Oliveiro
- **File:Blender255WorldContextButton.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/60/Blender255WorldContextButton.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* ?
- **File:Blender259BeachBall.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/66/Blender259BeachBall.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Falstaff
- **File:Blender259BeachBall_Selected.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/77/Blender259BeachBall_Selected.png *License:* GPL *Contributors:* Blender 2.59.0 *Original artist:* Blender Foundation
- **File:Blender259BeachBall_Yellow.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/44/Blender259BeachBall_Yellow.png *License:* GPL *Contributors:* Blender 2.59.0 *Original artist:* Blender Foundation

- **File:Blender259ObjectAddSecondMaterial.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/0f/Blender259ObjectAddSecondMaterial.png> License: GPL Contributors: Blender 2.59.0 Original artist: Blender Foundation
- **File:Blender259ObjectMaterial.png** Source: <https://upload.wikimedia.org/wikipedia/commons/9/9c/Blender259ObjectMaterial.png> License: GPL Contributors: Blender 2.59.0 Original artist: Blender Foundation
- **File:Blender259ObjectMaterialMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/1e/Blender259ObjectMaterialMenu.png> License: GPL Contributors: Blender 2.59.0 Original artist: Blender Foundation
- **File:Blender259ObjectMaterialUsers.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/54/Blender259ObjectMaterialUsers.png> License: GPL Contributors: Blender 2.59.0 Original artist: Blender Foundation
- **File:Blender259WoodTextureSample.png** Source: <https://upload.wikimedia.org/wikipedia/commons/2/26/Blender259WoodTextureSample.png> License: CC BY-SA 3.0 Contributors: Own work Original artist: Falstaff
- **File:Blender259_Texture_Button.png** Source: https://upload.wikimedia.org/wikipedia/commons/a/a4/Blender259_Texture_Button.png License: GPL Contributors: Blender 2.59.0 Original artist: Blender Foundation
- **File:Blender263AddMeshSubmenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/16/Blender263AddMeshSubmenu.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263CorrectiveShapeKeysEnabled.png** Source: <https://upload.wikimedia.org/wikipedia/commons/c/c8/Blender263CorrectiveShapeKeysEnabled.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263DefaultRender.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/19/Blender263DefaultRender.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263EditSelections.png** Source: <https://upload.wikimedia.org/wikipedia/commons/9/93/Blender263EditSelections.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263InitialPhysicsTab.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/13/Blender263InitialPhysicsTab.png> License: CC0 Contributors: Blender 2.63 Original artist: ?
- **File:Blender263LowPolyUVSphere.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/df/Blender263LowPolyUVSphere.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263LowPolyUVSphereFlat.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/fd/Blender263LowPolyUVSphereFlat.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263LowPolyUVSphereNormals.png** Source: <https://upload.wikimedia.org/wikipedia/commons/a/ae/Blender263LowPolyUVSphereNormals.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263LowPolyUVSphereSmooth.png** Source: <https://upload.wikimedia.org/wikipedia/commons/2/26/Blender263LowPolyUVSphereSmooth.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263LowPolyUVSphereSmoothGlitch.png** Source: <https://upload.wikimedia.org/wikipedia/commons/2/23/Blender263LowPolyUVSphereSmoothGlitch.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263ManipulatorOrientations.png** Source: <https://upload.wikimedia.org/wikipedia/commons/3/37/Blender263ManipulatorOrientations.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263ManipulatorRotate.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/81/Blender263ManipulatorRotate.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263ManipulatorScale.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/8b/Blender263ManipulatorScale.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263ManipulatorTranslate.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f1/Blender263ManipulatorTranslate.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263MeshDeleteMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/58/Blender263MeshDeleteMenu.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263MeshDisplayNormalsSettings.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/e7/Blender263MeshDisplayNormalsSettings.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263MeshEdgeDeleteAfter.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/d8/Blender263MeshEdgeDeleteAfter.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263MeshEdgeDeleteBefore.png** Source: <https://upload.wikimedia.org/wikipedia/commons/4/47/Blender263MeshEdgeDeleteBefore.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263MeshFaceDeleteAfter.png** Source: <https://upload.wikimedia.org/wikipedia/commons/4/45/Blender263MeshFaceDeleteAfter.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263MeshFaceDeleteBefore.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/0c/Blender263MeshFaceDeleteBefore.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263MeshVertexDeleteAfter.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/71/Blender263MeshVertexDeleteAfter.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263MeshVertexDeleteBefore.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/15/Blender263MeshVertexDeleteBefore.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263ModifiersMenuSubsurf.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f6/Blender263ModifiersMenuSubsurf.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263PivotPointMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/5c/Blender263PivotPointMenu.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263RenderDimensions.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/06/Blender263RenderDimensions.png> License: GPL Contributors: Blender 2.63 Original artist: ?

- **File:Blender263RenderDisplayMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/77/Blender263RenderDisplayMenu.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263RenderFormats.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/d5/Blender263RenderFormats.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263RenderRender.png** Source: <https://upload.wikimedia.org/wikipedia/commons/2/20/Blender263RenderRender.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263RendererSelectionMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/71/Blender263RendererSelectionMenu.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263SaveAsShapeKey.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/88/Blender263SaveAsShapeKey.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263ShadelessRender.png** Source: <https://upload.wikimedia.org/wikipedia/commons/9/98/Blender263ShadelessRender.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263ShadingButtons.png** Source: <https://upload.wikimedia.org/wikipedia/commons/4/41/Blender263ShadingButtons.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263SoftenedDefaultRender.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/eb/Blender263SoftenedDefaultRender.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263SubsurfModifierSettings.png** Source: <https://upload.wikimedia.org/wikipedia/commons/9/9f/Blender263SubsurfModifierSettings.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263SubsurfedCubeEditMode.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f2/Blender263SubsurfedCubeEditMode.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263SubsurfedCubeEditModeApplied.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/e9/Blender263SubsurfedCubeEditModeApplied.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263SubsurfedCubeLoopCut1.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/6d/Blender263SubsurfedCubeLoopCut1.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263SubsurfedCubeLoopCut2.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/7e/Blender263SubsurfedCubeLoopCut2.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263SubsurfedCubeLoopCut3.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/72/Blender263SubsurfedCubeLoopCut3.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263SubsurfedCubeObjectMode.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/02/Blender263SubsurfedCubeObjectMode.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263TextureIconWorld.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/ee/Blender263TextureIconWorld.png> License: CC0 Contributors: Blender 2.63 Original artist: ?
- **File:Blender263Timeline.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/e1/Blender263Timeline.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263TransformPanel.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/bc/Blender263TransformPanel.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender263WorldEnvironmentLightingSettings.png** Source: <https://upload.wikimedia.org/wikipedia/commons/9/96/Blender263WorldEnvironmentLightingSettings.png> License: GPL Contributors: Blender 2.63 Original artist: ?
- **File:Blender266FirstParticlesMaterial.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/70/Blender266FirstParticlesMaterial.png> License: GPL Contributors: Blender 2.66 Original artist: ?
- **File:Blender266ParticlePropsEmpty.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/d1/Blender266ParticlePropsEmpty.png> License: GPL Contributors: Blender 2.66 Original artist: ?
- **File:Blender266ParticlePropsInitial.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/82/Blender266ParticlePropsInitial.png> License: GPL Contributors: Blender 2.66 Original artist: ?
- **File:Blender266ParticlesFalling.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/b7/Blender266ParticlesFalling.png> License: GPL Contributors: Blender 2.66 Original artist: ?
- **File:Blender267BasicRayTraceArraySettings.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/e3/Blender267BasicRayTraceArraySettings.png> License: GPL Contributors: Blender 2.67 Original artist: ?
- **File:Blender267BasicRayTraceMaterialMirrorSettings.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/09/Blender267BasicRayTraceMaterialMirrorSettings.png> License: GPL Contributors: Blender 2.67 Original artist: ?
- **File:Blender267BasicRayTraceMaterialShadowSettings.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/b0/Blender267BasicRayTraceMaterialShadowSettings.png> License: GPL Contributors: Blender 2.67 Original artist: ?
- **File:Blender267BasicRayTraceMaterialTransparencySettings.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/7b/Blender267BasicRayTraceMaterialTransparencySettings.png> License: GPL Contributors: Blender 2.67 Original artist: ?
- **File:Blender267BasicRayTraceRender1.png** Source: <https://upload.wikimedia.org/wikipedia/commons/c/c4/Blender267BasicRayTraceRender1.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender267BasicRayTraceRender2.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/6b/Blender267BasicRayTraceRender2.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender267BasicRayTraceRender3.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/14/Blender267BasicRayTraceRender3.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender267BasicRayTraceRender4.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/58/Blender267BasicRayTraceRender4.png> License: CC0 Contributors: Own work Original artist: Ldo

- **File:Blender267BasicRayTraceRender5.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/e9/Blender267BasicRayTraceRender5.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender267BasicRayTraceRender6.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/8f/Blender267BasicRayTraceRender6.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender267BasicRayTraceShadows1.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f8/Blender267BasicRayTraceShadows1.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender267BasicRayTraceShadows2.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/67/Blender267BasicRayTraceShadows2.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender267BasicRayTraceWorldTextureSettings.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/64/Blender267BasicRayTraceWorldTextureSettings.png> License: GPL Contributors: Blender 2.67 Original artist: ?
- **File:Blender267CameraDataContext.png** Source: <https://upload.wikimedia.org/wikipedia/commons/4/4d/Blender267CameraDataContext.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267CameraDataContextButton.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/e5/Blender267CameraDataContextButton.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267CameraLensAngle.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/d9/Blender267CameraLensAngle.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267CurveDataContextBezier3D.png** Source: <https://upload.wikimedia.org/wikipedia/commons/4/4a/Blender267CurveDataContextBezier3D.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267CurveDataContextButton.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/1a/Blender267CurveDataContextButton.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267CyclesMaterialBSDFMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/c/cc/Blender267CyclesMaterialBSDFMenu.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender267CyclesMaterialContext.png** Source: <https://upload.wikimedia.org/wikipedia/commons/3/35/Blender267CyclesMaterialContext.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender267CyclesMaterialDiffuseBSDF.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/ec/Blender267CyclesMaterialDiffuseBSDF.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender267CyclesMaterialEmissionBSDF.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/84/Blender267CyclesMaterialEmissionBSDF.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender267CyclesRenderingContextSamplingPanel.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/b1/Blender267CyclesRenderingContextSamplingPanel.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender267CylinderEditMode.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/b8/Blender267CylinderEditMode.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender267DefaultCompositingNodes.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/67/Blender267DefaultCompositingNodes.png> License: GPL Contributors: Blender 2.66.6 Original artist: ?
- **File:Blender267DefaultCube.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/01/Blender267DefaultCube.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender267DefaultCubeSSS.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/56/Blender267DefaultCubeSSS.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender267DefaultCubeSSSR.png** Source: <https://upload.wikimedia.org/wikipedia/commons/2/28/Blender267DefaultCubeSSSR.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender267DeformModifierSettings.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/5b/Blender267DeformModifierSettings.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267DiceFirstLoopToRemove.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/b2/Blender267DiceFirstLoopToRemove.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267GlowingCubeCompositingNodes.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/6d/Blender267GlowingCubeCompositingNodes.png> License: GPL Contributors: Blender 2.66.6 Original artist: ?
- **File:Blender267GridPlusPath.png** Source: <https://upload.wikimedia.org/wikipedia/commons/9/97/Blender267GridPlusPath.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267GridPlusPathDeform.png** Source: <https://upload.wikimedia.org/wikipedia/commons/a/a9/Blender267GridPlusPathDeform.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267GridPlusPathDeform2.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f5/Blender267GridPlusPathDeform2.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267HalfPool.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/64/Blender267HalfPool.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender267HouseAddingArrayModifier.png** Source: <https://upload.wikimedia.org/wikipedia/commons/a/a4/Blender267HouseAddingArrayModifier.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender267HouseArrayModifier.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/df/Blender267HouseArrayModifier.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender267HouseBoxSelectWindowFrame.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/7b/Blender267HouseBoxSelectWindowFrame.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender267HouseCameraView.png** Source: <https://upload.wikimedia.org/wikipedia/commons/9/9e/Blender267HouseCameraView.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?

- **File:Blender267HouseColoured.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c7/Blender267HouseColoured.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseExtrudeWindowFrame.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/17/Blender267HouseExtrudeWindowFrame.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseExtrudeWindowFrame2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/81/Blender267HouseExtrudeWindowFrame2.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseFenceHoriz1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e0/Blender267HouseFenceHoriz1.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseFenceHoriz2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d4/Blender267HouseFenceHoriz2.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseFencePalings2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ad/Blender267HouseFencePalings2.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseFencePalings3.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5b/Blender267HouseFencePalings3.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseFencePalings4.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b2/Blender267HouseFencePalings4.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseFencePalings5.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/7d/Blender267HouseFencePalings5.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseGlassMaterial.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/08/Blender267HouseGlassMaterial.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseLighting.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b2/Blender267HouseLighting.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HousePath1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/93/Blender267HousePath1.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HousePath2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d7/Blender267HousePath2.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HousePath4.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/64/Blender267HousePath4.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HousePlusChimney.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/87/Blender267HousePlusChimney.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HousePlusWindow.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e6/Blender267HousePlusWindow.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseRoofApex.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ab/Blender267HouseRoofApex.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseRoofCube.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2e/Blender267HouseRoofCube.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseRoofCubeEdit.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/38/Blender267HouseRoofCubeEdit.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseRoofEaves.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e1/Blender267HouseRoofEaves.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseRoofMaterial.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e6/Blender267HouseRoofMaterial.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseWallsMaterial.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/00/Blender267HouseWallsMaterial.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267HouseWindowGlassColour.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/96/Blender267HouseWindowGlassColour.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267InitialBezierCurve.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c6/Blender267InitialBezierCurve.png> *License:* GPL *Contributors:* Blender 2.67a *Original artist:* ?
- **File:Blender267MaterialSSSPresets.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/78/Blender267MaterialSSSPresets.png> *License:* GPL *Contributors:* Blender 2.67 *Original artist:* ?
- **File:Blender267MaterialSSSSettings.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/05/Blender267MaterialSSSSettings.png> *License:* GPL *Contributors:* Blender 2.67 *Original artist:* ?
- **File:Blender267MaterialsPopupIcon.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/48/Blender267MaterialsPopupIcon.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267MeshDataContextButton.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/af/Blender267MeshDataContextButton.png> *License:* GPL *Contributors:* Blender 2.67.1 *Original artist:* ?
- **File:Blender267NURBSPatchExtrudeRowOfPoints.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/35/Blender267NURBSPatchExtrudeRowOfPoints.png> *License:* GPL *Contributors:* Blender 2.67a *Original artist:* ?
- **File:Blender267NURBSPatchSelectRowOfPoints.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5f/Blender267NURBSPatchSelectRowOfPoints.png> *License:* GPL *Contributors:* Blender 2.67a *Original artist:* ?
- **File:Blender267NURBSPatchSelectTwoRowsOfPoints.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/15/Blender267NURBSPatchSelectTwoRowsOfPoints.png> *License:* GPL *Contributors:* Blender 2.67a *Original artist:* ?

- **File:Blender267NURBSPatchSubdivideRowsOfPoints.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/69/Blender267NURBSPatchSubdivideRowsOfPoints.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267NURBSPointTransform.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/b5/Blender267NURBSPointTransform.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267NodeEditorTypeIcons.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/02/Blender267NodeEditorTypeIcons.png> License: GPL Contributors: Blender 2.66.6 Original artist: ?
- **File:Blender267ObjectContextButtons.png** Source: <https://upload.wikimedia.org/wikipedia/commons/a/aa/Blender267ObjectContextButtons.png> License: GPL Contributors: Blender 2.66.6 + Gimp Original artist: ?
- **File:Blender267ParachuteDeleteCylinderBottom.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/1e/Blender267ParachuteDeleteCylinderBottom.png> License: GPL Contributors: Blender 2.67.1 + Gimp Original artist: ?
- **File:Blender267PropertiesTabs.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/dc/Blender267PropertiesTabs.png> License: GPL Contributors: Blender 2.66.6 Original artist: ?
- **File:Blender267RainbowTextureNodes.png** Source: <https://upload.wikimedia.org/wikipedia/commons/c/cd/Blender267RainbowTextureNodes.png> License: GPL Contributors: Blender 2.66.6 Original artist: ?
- **File:Blender267RenderLayersContextButton.png** Source: <https://upload.wikimedia.org/wikipedia/commons/3/36/Blender267RenderLayersContextButton.png> License: GPL Contributors: Blender 2.66.6 Original artist: ?
- **File:Blender267SceneIcon.png** Source: <https://upload.wikimedia.org/wikipedia/commons/3/3a/Blender267SceneIcon.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender267ScreenLayoutIcon.png** Source: <https://upload.wikimedia.org/wikipedia/commons/9/9f/Blender267ScreenLayoutIcon.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender267SimpleLeaf.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/11/Blender267SimpleLeaf.png> License: GPL Contributors: Blender 2.67 Original artist: ?
- **File:Blender267Split3DView.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/60/Blender267Split3DView.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender267SubsurfedCubeCreasedEdges.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/19/Blender267SubsurfedCubeCreasedEdges.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267SurfaceDataContextButton.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/76/Blender267SurfaceDataContextButton.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267SurfaceDataContextNURBS.png** Source: <https://upload.wikimedia.org/wikipedia/commons/4/49/Blender267SurfaceDataContextNURBS.png> License: GPL Contributors: Blender 2.67a Original artist: ?
- **File:Blender267TextureCoordsSeparate.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/be/Blender267TextureCoordsSeparate.png> License: GPL Contributors: Blender 2.66.6 Original artist: ?
- **File:Blender267ViewportShadingCycles.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/51/Blender267ViewportShadingCycles.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender268ArrayModifierWithEmpty1.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/80/Blender268ArrayModifierWithEmpty1.png> License: GPL Contributors: Blender 2.68.5 Original artist: ?
- **File:Blender268CirclePlusIcon.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/8f/Blender268CirclePlusIcon.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268ClearParentMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/3/33/Blender268ClearParentMenu.png> License: GPL Contributors: Blender 2.68.5 Original artist: ?
- **File:Blender268CustomShaderNodeGroup1.png** Source: <https://upload.wikimedia.org/wikipedia/commons/c/cb/Blender268CustomShaderNodeGroup1.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CustomShaderNodeGroup2.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/7b/Blender268CustomShaderNodeGroup2.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CustomShaderNodeGroup3.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/84/Blender268CustomShaderNodeGroup3.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CustomShaderNodeGroup4.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/db/Blender268CustomShaderNodeGroup4.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CustomShaderNodeGroupResult.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/6e/Blender268CustomShaderNodeGroupResult.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesCompositingSimpleDespeckle.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/0b/Blender268CyclesCompositingSimpleDespeckle.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesDefaultCompositingNodes.png** Source: <https://upload.wikimedia.org/wikipedia/commons/3/34/Blender268CyclesDefaultCompositingNodes.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesDefaultCompositingNodesPlusLightingPasses.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/87/Blender268CyclesDefaultCompositingNodesPlusLightingPasses.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesFirefliesExample1.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/0d/Blender268CyclesFirefliesExample1.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesFirefliesExample2.png** Source: <https://upload.wikimedia.org/wikipedia/commons/a/a9/Blender268CyclesFirefliesExample2.png> License: GPL Contributors: Blender 2.68 Original artist: ?

- **File:Blender268CyclesFirefliesExample3.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/5f/Blender268CyclesFirefliesExample3.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesFirefliesExample4.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/fe/Blender268CyclesFirefliesExample4.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesFirefliesExample5.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/57/Blender268CyclesFirefliesExample5.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesFirefliesExample6.png** Source: <https://upload.wikimedia.org/wikipedia/commons/4/43/Blender268CyclesFirefliesExample6.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender268CyclesFirefliesExample7.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/8b/Blender268CyclesFirefliesExample7.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender268CyclesFirefliesPassesFixup1.png** Source: <https://upload.wikimedia.org/wikipedia/commons/2/28/Blender268CyclesFirefliesPassesFixup1.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesFirefliesPassesFixup2.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/0d/Blender268CyclesFirefliesPassesFixup2.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesFirefliesPassesFixup3.png** Source: <https://upload.wikimedia.org/wikipedia/commons/3/36/Blender268CyclesFirefliesPassesFixup3.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesFirefliesRenderSettings.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/04/Blender268CyclesFirefliesRenderSettings.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesGlassRender1.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/bc/Blender268CyclesGlassRender1.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender268CyclesGlassRender2.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/d7/Blender268CyclesGlassRender2.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender268CyclesLightPassesRecreationNaive.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/01/Blender268CyclesLightPassesRecreationNaive.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesLightPassesRecreationPart1.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/67/Blender268CyclesLightPassesRecreationPart1.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesLightPassesRecreationPart2.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f3/Blender268CyclesLightPassesRecreationPart2.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesLightPassesRecreationPart3.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/80/Blender268CyclesLightPassesRecreationPart3.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesLightPassesRecreationPart4.png** Source: <https://upload.wikimedia.org/wikipedia/commons/3/3a/Blender268CyclesLightPassesRecreationPart4.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesLightPassesRecreationPart5.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/15/Blender268CyclesLightPassesRecreationPart5.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesLightPassesRecreationPart6.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/15/Blender268CyclesLightPassesRecreationPart6.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesWorldNotUsingNodes.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/52/Blender268CyclesWorldNotUsingNodes.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268CyclesWorldUsingNodes.png** Source: <https://upload.wikimedia.org/wikipedia/commons/a/a7/Blender268CyclesWorldUsingNodes.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268HousePlusDoor.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/5f/Blender268HousePlusDoor.png> License: GPL Contributors: Blender 2.67.1 Original artist: ?
- **File:Blender268ImageEditorEditingMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/09/Blender268ImageEditorEditingMenu.png> License: GPL Contributors: Blender 2.68.5 Original artist: ?
- **File:Blender268ImageEditorImageMenus.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/b4/Blender268ImageEditorImageMenus.png> License: GPL Contributors: Blender 2.68.5 Original artist: ?
- **File:Blender268ImageEditorLayerMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/9/9d/Blender268ImageEditorLayerMenu.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268ImageEditorRenderResultMenus.png** Source: <https://upload.wikimedia.org/wikipedia/commons/9/90/Blender268ImageEditorRenderResultMenus.png> License: GPL Contributors: Blender 2.68.5 Original artist: ?
- **File:Blender268ImageEditorUVMenus.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/73/Blender268ImageEditorUVMenus.png> License: GPL Contributors: Blender 2.68.5 Original artist: ?
- **File:Blender268MonkeyBacklightPosition.png** Source: <https://upload.wikimedia.org/wikipedia/commons/2/2f/Blender268MonkeyBacklightPosition.png> License: GPL Contributors: Blender 2.68.5 Original artist: ?
- **File:Blender268NodeEditorShaderObjectDefault.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/09/Blender268NodeEditorShaderObjectDefault.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268NodeEditorShaderObjectGlass.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f5/Blender268NodeEditorShaderObjectGlass.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268NodeEditorShaderObjectGlass2.png** Source: <https://upload.wikimedia.org/wikipedia/commons/4/40/Blender268NodeEditorShaderObjectGlass2.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268NodeEditorShaderTypeIcons.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/82/Blender268NodeEditorShaderTypeIcons.png> License: GPL Contributors: Blender 2.68 Original artist: ?

- **File:Blender268NodeEditorShaderWorld+Magic.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/b2/Blender268NodeEditorShaderWorld%2BMagic.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268NodeEditorShaderWorldDefault.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/ee/Blender268NodeEditorShaderWorldDefault.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268NodeEditorTypeIcons.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/6c/Blender268NodeEditorTypeIcons.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268ObjectAddEmptyMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/db/Blender268ObjectAddEmptyMenu.png> License: GPL Contributors: Blender 2.68.5 Original artist: ?
- **File:Blender268ObjectContextTransformPanel.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f7/Blender268ObjectContextTransformPanel.png> License: GPL Contributors: Blender 2.68.5 Original artist: ?
- **File:Blender268PropertiesShelfTransformPanel.png** Source: <https://upload.wikimedia.org/wikipedia/commons/3/31/Blender268PropertiesShelfTransformPanel.png> License: GPL Contributors: Blender 2.68.5 Original artist: ?
- **File:Blender268RenderContextPostProcOptions.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/08/Blender268RenderContextPostProcOptions.png> License: GPL Contributors: Blender 2.68.5 Original artist: ?
- **File:Blender268RenderContextStampOptions.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/06/Blender268RenderContextStampOptions.png> License: GPL Contributors: Blender 2.68.5 Original artist: ?
- **File:Blender268RenderLayersContextDefault.png** Source: <https://upload.wikimedia.org/wikipedia/commons/4/42/Blender268RenderLayersContextDefault.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268SetParentMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/6c/Blender268SetParentMenu.png> License: GPL Contributors: Blender 2.68.5 Original artist: ?
- **File:Blender268ShaderNodeEditorAddGroupMenu+CustomGlass.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/b1/Blender268ShaderNodeEditorAddGroupMenu%2BCustomGlass.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268ShaderNodeEditorAddInputMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f5/Blender268ShaderNodeEditorAddInputMenu.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268ShaderNodeEditorAddShaderMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f6/Blender268ShaderNodeEditorAddShaderMenu.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268ShaderNodeEditorAddTextureMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/19/Blender268ShaderNodeEditorAddTextureMenu.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268SimpleDefaultTextureRender.png** Source: <https://upload.wikimedia.org/wikipedia/commons/c/c1/Blender268SimpleDefaultTextureRender.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender268SimpleDefaultTextureRender2.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/ef/Blender268SimpleDefaultTextureRender2.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender268SimpleDefaultTextureRender3.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/d0/Blender268SimpleDefaultTextureRender3.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender268SimpleDefaultTextureRender4.png** Source: <https://upload.wikimedia.org/wikipedia/commons/9/9d/Blender268SimpleDefaultTextureRender4.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender268SimpleDefaultTextureRender5.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/e7/Blender268SimpleDefaultTextureRender5.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender268TextureCategoryIcons.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f7/Blender268TextureCategoryIcons.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268TextureContext.png** Source: <https://upload.wikimedia.org/wikipedia/commons/2/27/Blender268TextureContext.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268TextureContextImagePanelNone.png** Source: <https://upload.wikimedia.org/wikipedia/commons/c/c0/Blender268TextureContextImagePanelNone.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268TextureContextImageSettings.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/5b/Blender268TextureContextImageSettings.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268TextureContextWorld.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/01/Blender268TextureContextWorld.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268TextureIconBrush.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f4/Blender268TextureIconBrush.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268TextureIconMaterial.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/df/Blender268TextureIconMaterial.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268TextureIconWorld.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/dd/Blender268TextureIconWorld.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender268TextureTypeMenuItemImage.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/79/Blender268TextureTypeMenuItemImage.png> License: GPL Contributors: Blender 2.68 Original artist: ?
- **File:Blender2693DViewIcon.png** Source: <https://upload.wikimedia.org/wikipedia/commons/2/2b/Blender2693DViewIcon.png> License: GPL Contributors: Blender 2.69.1 Original artist: ?
- **File:Blender269DefaultView.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/6a/Blender269DefaultView.png> License: GPL Contributors: Blender 2.69 Original artist: ?
- **File:Blender269HeaderMenu.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/fd/Blender269HeaderMenu.png> License: GPL Contributors: Blender 2.69 Original artist: ?

- **File:Blender269InfoIcon.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f9/Blender269InfoIcon.png> *License:* GPL
Contributors: Blender 2.69.1 *Original artist:* ?
- **File:Blender269NodeEditorIcon.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b1/Blender269NodeEditorIcon.png> *License:* GPL
Contributors: Blender 2.69.1 *Original artist:* ?
- **File:Blender269OutlinerIcon.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/85/Blender269OutlinerIcon.png> *License:* GPL
Contributors: Blender 2.69.1 *Original artist:* ?
- **File:Blender269PropertiesIcon.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/fd/Blender269PropertiesIcon.png> *License:* GPL
Contributors: Blender 2.69.1 *Original artist:* ?
- **File:Blender269PythonConsoleIcon.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/6d/Blender269PythonConsoleIcon.png> *License:* GPL
Contributors: Blender 2.69.1 *Original artist:* ?
- **File:Blender269TextEditorIcon.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/22/Blender269TextEditorIcon.png> *License:* GPL
Contributors: Blender 2.69.1 *Original artist:* ?
- **File:Blender269UVImageEditorIcon.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/17/Blender269UVImageEditorIcon.png> *License:* GPL
Contributors: Blender 2.69.1 *Original artist:* ?
- **File:Blender269UserPrefsIcon.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/56/Blender269UserPrefsIcon.png> *License:* GPL
Contributors: Blender 2.69.1 *Original artist:* ?
- **File:Blender269_3DViewWindowType.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/68/Blender269_3DViewWindowType.png *License:* GPL
Contributors: Blender 2.69 *Original artist:* ?
- **File:Blender270BackgroundImages1Image.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2b/Blender270BackgroundImages1Image.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270BackgroundImages1ImageLoaded.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1a/Blender270BackgroundImages1ImageLoaded.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270BackgroundImagesAxisMenu.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/50/Blender270BackgroundImagesAxisMenu.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270BackgroundImagesDefault.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/42/Blender270BackgroundImagesDefault.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270BezierBevel0.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f7/Blender270BezierBevel0.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270BezierBevel1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9f/Blender270BezierBevel1.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270BezierBevel2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5c/Blender270BezierBevel2.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270BezierBevel3.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/42/Blender270BezierBevel3.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270BezierBevel4.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/19/Blender270BezierBevel4.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270BezierBevel5.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/95/Blender270BezierBevel5.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270BezierBevel6.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/23/Blender270BezierBevel6.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270BezierBevel7.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/17/Blender270BezierBevel7.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270BezierBevel8.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/fa/Blender270BezierBevel8.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270BezierBevelCustom1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/01/Blender270BezierBevelCustom1.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270BoneContextButton.PNG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/92/Blender270BoneContextButton.PNG> *License:* CC BY-SA 4.0
Contributors: Own work *Original artist:* TROPtastic
- **File:Blender270DopeSheetIcon.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/72/Blender270DopeSheetIcon.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270GraphEditorIcon.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ad/Blender270GraphEditorIcon.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270GraphEditorPivotPointMenu.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/79/Blender270GraphEditorPivotPointMenu.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270GraphEditorSimpleCube1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b2/Blender270GraphEditorSimpleCube1.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270NLAEditorIcon.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5a/Blender270NLAEditorIcon.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270SquaringBezierCircle0.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1d/Blender270SquaringBezierCircle0.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?
- **File:Blender270SquaringBezierCircle1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/16/Blender270SquaringBezierCircle1.png> *License:* GPL
Contributors: Blender 2.70 *Original artist:* ?

- **File:Blender270SquaringBezierCircle2.png** Source: <https://upload.wikimedia.org/wikipedia/commons/a/ab/Blender270SquaringBezierCircle2.png> License: GPL Contributors: Blender 2.70 Original artist: ?
- **File:Blender270TimelineIcon.png** Source: <https://upload.wikimedia.org/wikipedia/commons/2/26/Blender270TimelineIcon.png> License: GPL Contributors: Blender 2.70 Original artist: ?
- **File:Blender270WorldExample1BlendSky.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/7f/Blender270WorldExample1BlendSky.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender270WorldExample1RealPaperSky.png** Source: <https://upload.wikimedia.org/wikipedia/commons/3/3d/Blender270WorldExample1RealPaperSky.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender270WorldExample1RealSky.png** Source: <https://upload.wikimedia.org/wikipedia/commons/4/4a/Blender270WorldExample1RealSky.png> License: CC0 Contributors: Own work Original artist: Ldo
- **File:Blender270WorldSettingsDefault.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/71/Blender270WorldSettingsDefault.png> License: GPL Contributors: Blender 2.70 Original artist: ?
- **File:Blender270WorldSettingsExample1.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/b2/Blender270WorldSettingsExample1.png> License: GPL Contributors: Blender 2.70 Original artist: ?
- **File:Blender274RevealShelfIconBrightened.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/86/Blender274RevealShelfIconBrightened.png> License: CC BY-SA 4.0 Contributors: Own work Original artist: Anodragon453
- **File:Blender278aQuickieTexFinalRender.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/d6/Blender278aQuickieTexFinalRender.png> License: GPL Contributors: Blender 2.78a Original artist: Blender 2.78a
- **File:Blender278aQuickieTextureContext.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/e6/Blender278aQuickieTextureContext.png> License: GPL Contributors: Blender 2.78a Original artist: Blender 2.78a
- **File:Blender278aRendered1.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/6a/Blender278aRendered1.png> License: GPL Contributors: Blender 2.78a Original artist: Blender 2.78a
- **File:Blender3D-2.5properties-modifier.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/ec/Blender3D-2.5properties-modifier.png> License: CC BY-SA 3.0 Contributors: Own work Original artist: Toadster
- **File:Blender3D-Jute_zugeschnitten.png** Source: https://upload.wikimedia.org/wikipedia/commons/e/ed/Blender3D-Jute_zugeschnitten.png License: CC-BY-SA-3.0 Contributors: Original artist: SoylentGreen
- **File:Blender3D-buttonsWindow-editButtons.jpg** Source: <https://upload.wikimedia.org/wikipedia/commons/c/ce/Blender3D-buttonsWindow-editButtons.jpg> License: Public domain Contributors: <http://en.wikibooks.org/wiki/File:Blender3D-buttonsWindow-editButtons.jpg> Original artist: b:en:User:Spiderworm
- **File:Blender3D-joiningWindows.jpg** Source: <https://upload.wikimedia.org/wikipedia/commons/e/ef/Blender3D-joiningWindows.jpg> License: Public domain Contributors: <http://en.wikibooks.org/wiki/File:Blender3D-joiningWindows.jpg> Original artist: b:en:User:Spiderworm
- **File:Blender3D-userPrefs-AutoSave.jpg** Source: <https://upload.wikimedia.org/wikipedia/commons/6/6d/Blender3D-userPrefs-AutoSave.jpg> License: Public domain Contributors: <http://en.wikibooks.org/wiki/File:Blender3D-userPrefs-AutoSave.jpg> Original artist: b:en:User:Spiderworm
- **File:Blender3D-userPrefs-SaveSettings.png** Source: <https://upload.wikimedia.org/wikipedia/commons/c/c1/Blender3D-userPrefs-SaveSettings.png> License: CC-BY-SA-3.0 Contributors: Own work Original artist: Yhevhe
- **File:Blender3DDarkShot_0013.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/d/d0/Blender3DDarkShot_0013.jpg License: CC-BY-SA-3.0 Contributors: ? Original artist: ?
- **File:Blender3DFeet-3.jpg** Source: <https://upload.wikimedia.org/wikipedia/commons/0/05/Blender3DFeet-3.jpg> License: CC-BY-SA-3.0 Contributors: ? Original artist: ?
- **File:Blender3DNobbToPro-Creating_the_Stencil_04.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/6/6e/Blender3DNobbToPro-Creating_the_Stencil_04.jpg License: CC-BY-SA-3.0 Contributors: Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. Original artist: Moohasha. Original uploader was Moohasha at en.wikibooks
- **File:Blender3DNobbToPro-ExampleHeightmap.jpg** Source: <https://upload.wikimedia.org/wikipedia/commons/0/01/Blender3DNobbToPro-ExampleHeightmap.jpg> License: CC-BY-SA-3.0 Contributors: Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. Original artist: Moohasha. Original uploader was Moohasha at en.wikibooks
- **File:Blender3DNobbToPro-Exporting_as_a_Heightmap01.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/9/9c/Blender3DNobbToPro-Exporting_as_a_Heightmap01.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: Moohasha
- **File:Blender3DNobbToPro-Exporting_as_a_Heightmap02.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/6/63/Blender3DNobbToPro-Exporting_as_a_Heightmap02.jpg License: GPL Contributors: Own work Original artist: Moohasha
- **File:Blender3DNobbToPro-Exporting_as_a_Heightmap03.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/2/29/Blender3DNobbToPro-Exporting_as_a_Heightmap03.jpg License: GPL Contributors: Own work Original artist: Moohasha
- **File:Blender3DNobbToPro-Exporting_as_a_Heightmap_04.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/5/5c/Blender3DNobbToPro-Exporting_as_a_Heightmap_04.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: Moohasha
- **File:Blender3DNobbToPro-Exporting_as_a_Heightmap_05.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/2/2f/Blender3DNobbToPro-Exporting_as_a_Heightmap_05.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: Moohasha
- **File:Blender3DNobbToPro-Exporting_as_a_Heightmap_06.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/e/e0/Blender3DNobbToPro-Exporting_as_a_Heightmap_06.jpg License: GPL Contributors: Own work Original artist: Moohasha

- **File:Blender3DNoobToPro-Applying_the_Stencil_01.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/c/cb/Blender3DNoobToPro-Applying_the_Stencil_01.jpg License: GPL Contributors: Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. Original artist: The original uploader was Moohasha at English Wikibooks
- **File:Blender3DNoobToPro-Applying_the_Stencil_02.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/9/98/Blender3DNoobToPro-Applying_the_Stencil_02.jpg License: Public domain Contributors: Own work Original artist: Moohasha. The original uploader was Moohasha at English Wikibooks
- **File:Blender3DNoobToPro-Creating_The_Canvas.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/1/16/Blender3DNoobToPro-Creating_The_Canvas.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: Moohasha
- **File:Blender3DNoobToPro-Creating_the_Stencil_02.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/c/ce/Blender3DNoobToPro-Creating_the_Stencil_02.jpg License: GPL Contributors: Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. Original artist: Original uploader was Moohasha at en.wikibooks
- **File:Blender3DNoobToPro-Creating_the_Stencil_03.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/9/90/Blender3DNoobToPro-Creating_the_Stencil_03.jpg License: CC-BY-SA-3.0 Contributors: Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. Original artist: Moohasha. Original uploader was Moohasha at en.wikibooks
- **File:Blender3DNoobToPro-FirstMountain.png** Source: <https://upload.wikimedia.org/wikipedia/commons/c/c6/Blender3DNoobToPro-FirstMountain.png> License: GPL Contributors: Transferred from en.wikibooks to Commons. Original artist: The original uploader was Megaloman at English Wikibooks
- **File:Blender3DNoobToPro-Grass.jpg** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f2/Blender3DNoobToPro-Grass.jpg> License: Public domain Contributors: Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. Original artist: Moohasha at English Wikibooks
- **File:Blender3DNoobToPro-Molding_the_Mountains_01.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/3/3e/Blender3DNoobToPro-Molding_the_Mountains_01.jpg License: GPL Contributors: Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. Original artist: The original uploader was Moohasha at English Wikibooks
- **File:Blender3DNoobToPro-Molding_the_Mountains_02.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/0/04/Blender3DNoobToPro-Molding_the_Mountains_02.jpg License: CC-BY-SA-3.0 Contributors: Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. Original artist: Moohasha. Original uploader was Moohasha at en.wikibooks
- **File:Blender3DNoobToPro-Molding_the_Mountains_03.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/5/53/Blender3DNoobToPro-Molding_the_Mountains_03.jpg License: CC-BY-SA-3.0 Contributors: Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. Original artist: Moohasha. Original uploader was Moohasha at en.wikibooks
- **File:Blender3DNoobToPro-Molding_the_Mountains_04.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/7/70/Blender3DNoobToPro-Molding_the_Mountains_04.jpg License: GPL Contributors: Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. Original artist: The original uploader was Moohasha at English Wikibooks
- **File:Blender3DNoobToPro-Molding_the_Mountains_05.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/7/71/Blender3DNoobToPro-Molding_the_Mountains_05.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: Moohasha
- **File:Blender3DNoobToPro-Multiple_Stencils_01.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/5/50/Blender3DNoobToPro-Multiple_Stencils_01.jpg License: CC BY-SA 3.0 Contributors: Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. Original artist: Moohasha. Original uploader was Moohasha at en.wikibooks
- **File:Blender3DNoobToPro-Multiple_Stencils_02.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/6/6d/Blender3DNoobToPro-Multiple_Stencils_02.jpg License: CC BY-SA 3.0 Contributors: Own work Original artist: Moohasha
- **File:Blender3DNoobToPro-Multiple_Stencils_03.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/1/10/Blender3DNoobToPro-Multiple_Stencils_03.jpg License: CC BY-SA 3.0 Contributors: Own work Original artist: Moohasha
- **File:Blender3DNoobToPro-Multiple_Stencils_04.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/3/36/Blender3DNoobToPro-Multiple_Stencils_04.jpg License: CC BY-SA 3.0 Contributors: Own work Original artist: Moohasha
- **File:Blender3DNoobToPro-RoundedMtns.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/59/Blender3DNoobToPro-RoundedMtns.png> License: GPL Contributors: Transferred from en.wikibooks Original artist: Original uploader was Megaloman at en.wikibooks
- **File:Blender3DNoobToPro-SharpVsSmoothFalloff.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/1a/Blender3DNoobToPro-SharpVsSmoothFalloff.png> License: GPL Contributors: Transferred from en.wikibooks Original artist: Original uploader was Megaloman at en.wikibooks
- **File:Blender3DNoobToPro-Sharpfalloff.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/5f/Blender3DNoobToPro-Sharpfalloff.png> License: GPL Contributors: Transferred from en.wikibooks to Commons. Original artist: The original uploader was Megaloman at English Wikibooks
- **File:Blender3DNoobToPro-SimpleHat-HatSpun.png** Source: <https://upload.wikimedia.org/wikipedia/commons/a/a5/Blender3DNoobToPro-SimpleHat-HatSpun.png> License: GPL Contributors: Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. Original artist: The original uploader was Wwwwolf at English Wikibooks
- **File:Blender3DNoobToPro-SimpleHat-Profile.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/d3/Blender3DNoobToPro-SimpleHat-Profile.png> License: GPL Contributors: Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. Original artist: The original uploader was Wwwwolf at English Wikibooks
- **File:Blender3DNoobToPro-SimplePlane.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/ec/Blender3DNoobToPro-SimplePlane.png> License: GPL Contributors: Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. Original artist: The original uploader was Megaloman at English Wikibooks

- **File:Blender3DNoobToPro-Texturing_the_Terrain_02.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/d/df/Blender3DNoobToPro-Texturing_the_Terrain_02.jpg License: GPL Contributors: Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. Original artist: The original uploader was Moohasha at English Wikibooks
- **File:Blender3DNoobToPro-Texturing_the_Terrain_03.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/0/0f/Blender3DNoobToPro-Texturing_the_Terrain_03.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: Moohasha
- **File:Blender3DNooptoPro-SharpMtnCreasing.PNG** Source: <https://upload.wikimedia.org/wikipedia/commons/c/c4/Blender3DNooptoPro-SharpMtnCreasing.PNG> License: GPL Contributors: Transferred from en.wikibooks Original artist: Original uploader was Skeep at en.wikibooks
- **File:Blender3D_2.4.7-screen.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/0/0d/Blender3D_2.4.7-screen.jpg License: CC-BY-SA-3.0 Contributors: Own work: The scene in the program is my own work. Original artist: Thomas Dinges
- **File:Blender3D_2.77_N2P_follow_Path.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/1/18/Blender3D_2.77_N2P_follow_Path.jpg License: GPL Contributors: Own work Original artist: Animajosser
- **File:Blender3D_2.77_N2P_follow_Path_2.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/e/eb/Blender3D_2.77_N2P_follow_Path_2.jpg License: GPL Contributors: Own work Original artist: Animajosser
- **File:Blender3D_2MaterialsSameTexture_2.46.png** Source: https://upload.wikimedia.org/wikipedia/commons/0/0e/Blender3D_2MaterialsSameTexture_2.46.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_2ObjectsSameMesh_2.46.png** Source: https://upload.wikimedia.org/wikipedia/commons/7/77/Blender3D_2ObjectsSameMesh_2.46.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_3TubesJoint.png** Source: https://upload.wikimedia.org/wikipedia/commons/c/c7/Blender3D_3TubesJoint.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_6TubesJoint.png** Source: https://upload.wikimedia.org/wikipedia/commons/a/a2/Blender3D_6TubesJoint.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_AddAParticleSystem-2.48a.png** Source: https://upload.wikimedia.org/wikipedia/commons/e/e7/Blender3D_AddAParticleSystem-2.48a.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_AddingAParticleSystem.png** Source: https://upload.wikimedia.org/wikipedia/commons/4/41/Blender3D_AddingAParticleSystem.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_AligningTwoCubes.gif** Source: https://upload.wikimedia.org/wikipedia/commons/c/cd/Blender3D_AligningTwoCubes.gif License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_AligningTwoObjects.gif** Source: https://upload.wikimedia.org/wikipedia/commons/e/ea/Blender3D_AligningTwoObjects.gif License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_AnimatedBumpMapApplied0022-2.49.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/2/25/Blender3D_AnimatedBumpMapApplied0022-2.49.jpg License: Public domain Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_BKFootRig2.png** Source: https://upload.wikimedia.org/wikipedia/commons/f/f0/Blender3D_BKFootRig2.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: Soylentgreen
- **File:Blender3D_BW-Testgrid-Applied.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/e/ee/Blender3D_BW-Testgrid-Applied.jpg License: Public domain Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_BW_Grid_256.png** Source: https://upload.wikimedia.org/wikipedia/commons/4/45/Blender3D_BW_Grid_256.png License: Public domain Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_BillboardAnimation-FlippedBlend.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/c/c8/Blender3D_BillboardAnimation-FlippedBlend.jpg License: Public domain Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_BillboardAnimationColor-Frame-50.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/b/b3/Blender3D_BillboardAnimationColor-Frame-50.jpg License: Public domain Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_BillboardParticlesSplitSettings.png** Source: https://upload.wikimedia.org/wikipedia/commons/1/18/Blender3D_BillboardParticlesSplitSettings.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_BillboardUvSplit_Frame0041.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/2/25/Blender3D_BillboardUvSplit_Frame0041.jpg License: Public domain Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_BridgeFaces.png** Source: https://upload.wikimedia.org/wikipedia/commons/a/a8/Blender3D_BridgeFaces.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_BrownianMotionAnimation-2.48a.png** Source: https://upload.wikimedia.org/wikipedia/commons/8/85/Blender3D_BrownianMotionAnimation-2.48a.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_BumpMapSettings-2.49.png** Source: https://upload.wikimedia.org/wikipedia/commons/6/6c/Blender3D_BumpMapSettings-2.49.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_CHF_EmitterObject.png** Source: https://upload.wikimedia.org/wikipedia/commons/2/22/Blender3D_CHF_EmitterObject.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_ColorDirtmap-2.49.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/f/f1/Blender3D_ColorDirtmap-2.49.jpg License: GFDL Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_CreaseACube.png** Source: https://upload.wikimedia.org/wikipedia/commons/f/f6/Blender3D_CreaseACube.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_CreatingABumpMapCamera-2.49.png** Source: https://upload.wikimedia.org/wikipedia/commons/4/44/Blender3D_CreatingABumpMapCamera-2.49.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_CreatingABumpMapCompositeNodes-2.49.png** Source: https://upload.wikimedia.org/wikipedia/commons/3/31/Blender3D_CreatingABumpMapCompositeNodes-2.49.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen

- **File:Blender3D_CuttingThroughSteel-AlphaMap-0026.png** Source: https://upload.wikimedia.org/wikipedia/commons/c/c5/Blender3D_CuttingThroughSteel-AlphaMap-0026.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_CuttingThroughSteel-AlphaMapMaterial-2.48a.png** Source: https://upload.wikimedia.org/wikipedia/commons/6/61/Blender3D_CuttingThroughSteel-AlphaMapMaterial-2.48a.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_CuttingThroughSteel-AlphaMapView-2.48a.png** Source: https://upload.wikimedia.org/wikipedia/commons/a/a1/Blender3D_CuttingThroughSteel-AlphaMapView-2.48a.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_CuttingThroughSteel-AlphaParticleSystem-2.48a.png** Source: https://upload.wikimedia.org/wikipedia/commons/7/74/Blender3D_CuttingThroughSteel-AlphaParticleSystem-2.48a.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_CuttingThroughSteel-LampEnergieIpo-2.48a.png** Source: https://upload.wikimedia.org/wikipedia/commons/f/f1/Blender3D_CuttingThroughSteel-LampEnergieIpo-2.48a.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_CuttingThroughSteel-ObjectCamera-2.48a.png** Source: https://upload.wikimedia.org/wikipedia/commons/8/87/Blender3D_CuttingThroughSteel-ObjectCamera-2.48a.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_CuttingThroughSteel-PlaneMaterial-Frame34.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/3/35/Blender3D_CuttingThroughSteel-PlaneMaterial-Frame34.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_CuttingThroughSteel-Sparks-Frame34.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/2/28/Blender3D_CuttingThroughSteel-Sparks-Frame34.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_CuttingThroughSteel-SparksParticles-2.48a.png** Source: https://upload.wikimedia.org/wikipedia/commons/4/4b/Blender3D_CuttingThroughSteel-SparksParticles-2.48a.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_CuttingThroughSteel-SparksSmoke-Frame34.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/f/f3/Blender3D_CuttingThroughSteel-SparksSmoke-Frame34.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_CuttingThroughSteel-SparksSmokeGlow-Frame34.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/b/bd/Blender3D_CuttingThroughSteel-SparksSmokeGlow-Frame34.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_CuttingThroughSteel_Smoke.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/1/13/Blender3D_CuttingThroughSteel_Smoke.jpg License: Public domain Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_CuttingThroughSteel_GlowMap-34.png** Source: https://upload.wikimedia.org/wikipedia/commons/c/c0/Blender3D_CuttingThroughSteel_GlowMap-34.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_DiffuseMapSettings-2.49.png** Source: https://upload.wikimedia.org/wikipedia/commons/8/88/Blender3D_DiffuseMapSettings-2.49.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_DirectionOfRotation.png** Source: https://upload.wikimedia.org/wikipedia/commons/9/9d/Blender3D_DirectionOfRotation.png License: Public domain Contributors: Myself, based on work of Schorschi2. see <http://de.wikipedia.org/wiki/Benutzer:Schorschi2> Original artist: SoylentGreen
- **File:Blender3D_DisplacementHoehenfeld.png** Source: https://upload.wikimedia.org/wikipedia/commons/7/74/Blender3D_DisplacementHoehenfeld.png License: GFDL Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_Documentation_1.png** Source: https://upload.wikimedia.org/wikipedia/commons/d/d7/Blender3D_Documentation_1.png License: Public domain Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_DrawType.png** Source: https://upload.wikimedia.org/wikipedia/commons/c/c4/Blender3D_DrawType.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_File_menu_screenshot.png** Source: https://upload.wikimedia.org/wikipedia/commons/f/f0/Blender3D_File_menu_screenshot.png License: GPL Contributors: Blender 2.67b Original artist: Blender Foundation
- **File:Blender3D_Fireworks-Reactor1-2.48a.png** Source: https://upload.wikimedia.org/wikipedia/commons/5/55/Blender3D_Fireworks-Reactor1-2.48a.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_Fireworks-Reactor1-Material-2.48a.png** Source: https://upload.wikimedia.org/wikipedia/commons/a/a9/Blender3D_Fireworks-Reactor1-Material-2.48a.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_Fireworks-Reactor2-2.48a.png** Source: https://upload.wikimedia.org/wikipedia/commons/4/41/Blender3D_Fireworks-Reactor2-2.48a.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_Fireworks-Reactor2-Material-2.48a.png** Source: https://upload.wikimedia.org/wikipedia/commons/2/2d/Blender3D_Fireworks-Reactor2-Material-2.48a.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_FirstKeyedSystem-2.48a.png** Source: https://upload.wikimedia.org/wikipedia/commons/8/83/Blender3D_FirstKeyedSystem-2.48a.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: SoylentGreen
- **File:Blender3D_FlatShading.PNG** Source: https://upload.wikimedia.org/wikipedia/commons/1/16/Blender3D_FlatShading.PNG License: GPL Contributors: Screenshot in Blender3D (GPL) software Original artist: Jake.lewis4
- **File:Blender3D_FreeNote.png** Source: https://upload.wikimedia.org/wikipedia/commons/5/53/Blender3D_FreeNote.png License: CC-BY-SA-3.0 Contributors: Myself, based on an icon from the crystal icons of KDE. Original artist: SoylentGreen

- **File:Blender3D_FurTexture_for_FurTutorial.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a1/Blender3D_FurTexture_for_FurTutorial.png *License:* CC BY-SA 2.5 *Contributors:* Original artist: SoylentGreen
- **File:Blender3D_FurWithParticles-ButtonToClick-2.49a.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a5/Blender3D_FurWithParticles-ButtonToClick-2.49a.png *License:* GPL *Contributors:* Blender3D *Original artist:* Blender Foundation
- **File:Blender3D_FurWithParticles-ColorTexture-2.48a.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c5/Blender3D_FurWithParticles-ColorTexture-2.48a.jpg *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_FurWithParticles-Combing-2.48a.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/90/Blender3D_FurWithParticles-Combing-2.48a.png *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_FurWithParticles-Finished-2.48a.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/3b/Blender3D_FurWithParticles-Finished-2.48a.jpg *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_FurWithParticles-NoMaterial-2.48a.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6b/Blender3D_FurWithParticles-NoMaterial-2.48a.jpg *License:* Public domain *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_FurWithParticles-ParticleLengthSettings-2.48a.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/af/Blender3D_FurWithParticles-ParticleLengthSettings-2.48a.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_FurWithParticles-ParticleLengthTexture-2.48a.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/4b/Blender3D_FurWithParticles-ParticleLengthTexture-2.48a.jpg *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_FurWithParticles-TexturePainting-2.48a.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9a/Blender3D_FurWithParticles-TexturePainting-2.48a.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_Hair_Brushed.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9f/Blender3D_Hair_Brushed.png *License:* CC-BY-SA-3.0 *Contributors:* Screenshot of the GPL program Blender *Original artist:* WebBird
- **File:Blender3D_HarmonicFieldOnParticles.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/7e/Blender3D_HarmonicFieldOnParticles.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_HarmonicFieldParticlesNoObject.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0a/Blender3D_HarmonicFieldParticlesNoObject.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_KissingFaces3a_2.46.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8c/Blender3D_KissingFaces3a_2.46.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_KissingFaces5_2.46.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/79/Blender3D_KissingFaces5_2.46.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_KissingFaces6_2.46.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c4/Blender3D_KissingFaces6_2.46.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_KolbenZylinderAnimation.gif** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f5/Blender3D_KolbenZylinderAnimation.gif *License:* CC-BY-SA-3.0 *Contributors:* own work, basiert auf einem Modell (Airship) aus den Releasenotes *Original artist:* Soylentgreen
- **File:Blender3D_KolbenZylinderArmaturePositions.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/88/Blender3D_KolbenZylinderArmaturePositions.png *License:* CC-BY-SA-3.0 *Contributors:* own work, basiert auf einem Modell (Airship) aus den Releasenotes *Original artist:* Soylentgreen
- **File:Blender3D_KolbenZylinderBHelper1Constraint.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a7/Blender3D_KolbenZylinderBHelper1Constraint.png *License:* CC-BY-SA-3.0 *Contributors:* own work, basiert auf einem Modell (Airship) aus den Releasenotes *Original artist:* Soylentgreen
- **File:Blender3D_KolbenZylinderESteuerungConstraint.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/45/Blender3D_KolbenZylinderESteuerungConstraint.png *License:* CC-BY-SA-3.0 *Contributors:* own work, basiert auf einem Modell (Airship) aus den Releasenotes *Original artist:* Soylentgreen
- **File:Blender3D_KolbenZylinderEmptyPositions.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f0/Blender3D_KolbenZylinderEmptyPositions.png *License:* CC-BY-SA-3.0 *Contributors:* own work, basiert auf einem Modell (Airship) aus den Releasenotes *Original artist:* Soylentgreen
- **File:Blender3D_KolbenZylinderExample.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/69/Blender3D_KolbenZylinderExample.png *License:* CC-BY-SA-3.0 *Contributors:* own work, basiert auf einem Modell (Airship) aus den Releasenotes *Original artist:* Soylentgreen
- **File:Blender3D_Lagerfeuer_Textur.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/55/Blender3D_Lagerfeuer_Textur.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender. Copied from de.wikibooks. *Original artist:* Eilexe
- **File:Blender3D_LargeSmallTube.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/93/Blender3D_LargeSmallTube.png *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_LargeSmallTubeResult.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/ce/Blender3D_LargeSmallTubeResult.png *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_Limit_selection_to_visible.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e0/Blender3D_Limit_selection_to_visible.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Romain Behar
- **File:Blender3D_Lisc_lipy-Transparent.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/be/Blender3D_Lisc_lipy-Transparent.png *License:* CC-BY-SA-3.0 *Contributors:* Myself, based on file:Lisc lipy.jpg *Original artist:* SoylentGreen, Krzysztof P. Jasiutowicz

- **File:Blender3D_MakeVertexParentSchwungrad.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8c/Blender3D_MakeVertexParentSchwungrad.png *License:* CC-BY-SA-3.0 *Contributors:* own work, basiert auf einem Modell (Airship) aus den Releasenotes *Original artist:* Soylentgreen
- **File:Blender3D_MapToPanelStreichholz.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5c/Blender3D_MapToPanelStreichholz.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_MapleGreyscaleResult.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/21/Blender3D_MapleGreyscaleResult.jpg *License:* Public domain *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_MapleGreyscaleSettings.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/59/Blender3D_MapleGreyscaleSettings.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_Maple_ColorAlphaTexture_New.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/65/Blender3D_Maple_ColorAlphaTexture_New.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender / File:Blender3D Maple ColorAlphaTexture.png *Original artist:* SoylentGreen (Edited by:Weeix)
- **File:Blender3D_Maple_Greyscale.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/4c/Blender3D_Maple_Greyscale.png *License:* CC BY-SA 4.0-3.0-2.5-2.0-1.0 *Contributors:* Myself, based on an image texture by Gabio from Blender Artists.org *Original artist:* SoylentGreen
- **File:Blender3D_MultipleUV-Textures.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5d/Blender3D_MultipleUV-Textures.jpg *License:* CC BY 2.5 *Contributors:* Rendererd from the files of Elephants Dream. *Original artist:* SoylentGreen
- **File:Blender3D_ObjectBlockSelection_2.46.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b9/Blender3D_ObjectBlockSelection_2.46.png *License:* CC BY 2.5 *Contributors:* Myself, based on files from the film Elephants Dream, made by the Blender Foundation. *Original artist:* SoylentGreen
- **File:Blender3D_ObjectSnapping-2.46.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/1e/Blender3D_ObjectSnapping-2.46.png *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_Objectplan_2.46.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/54/Blender3D_Objectplan_2.46.png *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_ParentChildCoordinates.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/17/Blender3D_ParentChildCoordinates.png *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_ParticleMaterialForKeyedSystem-2.48a.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/52/Blender3D_ParticleMaterialForKeyedSystem-2.48a.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_ParticleSystemForFur-2.48a.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d6/Blender3D_ParticleSystemForFur-2.48a.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_PositioningWithTwoEmptys_F19-c.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/50/Blender3D_PositioningWithTwoEmptys_F19-c.png *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* Soylentgreen
- **File:Blender3D_PositioningWithTwoEmptys_F22- _a.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c4/Blender3D_PositioningWithTwoEmptys_F22- _a.png *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* Soylentgreen
- **File:Blender3D_RetopoPaintExample.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5c/Blender3D_RetopoPaintExample.png *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_Roehrenabzweig.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e4/Blender3D_Roehrenabzweig.png *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_Roehrenabzweig2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/23/Blender3D_Roehrenabzweig2.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Blender3D_SelectionModes.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5e/Blender3D_SelectionModes.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Argento at English Wikibooks
- **File:Blender3D_ShuffleStride4.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9c/Blender3D_ShuffleStride4.png *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* Soylentgreen
- **File:Blender3D_SimpleFireMaterial-2.48a.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f6/Blender3D_SimpleFireMaterial-2.48a.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_SimpleFireMaterialAnimation-2.48a.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b4/Blender3D_SimpleFireMaterialAnimation-2.48a.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_SimpleFireNoMaterial-2.48a.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/25/Blender3D_SimpleFireNoMaterial-2.48a.jpg *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_SimpleFireParticleSystem-2.48a.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5c/Blender3D_SimpleFireParticleSystem-2.48a.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_SimpleFireSetup-2.48a.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/90/Blender3D_SimpleFireSetup-2.48a.jpg *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_SimplePerson_Scale_Pelvis.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c8/Blender3D_SimplePerson_Scale_Pelvis.png *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Bgks at English Wikibooks
- **File:Blender3D_SimplePerson_Snap_Pelvis.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/07/Blender3D_SimplePerson_Snap_Pelvis.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Bgks at English Wikibooks

- **File:Blender3D_SmoothShading.PNG** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/41/Blender3D_SmoothShading.PNG *License:* GPL *Contributors:* Screenshot in Blender3D (GPL) software *Original artist:* Jake.lewis4
- **File:Blender3D_StencilDirtMapSettings-2.49.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f2/Blender3D_StencilDirtMapSettings-2.49.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_StencilingTexture-Example.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/99/Blender3D_StencilingTexture-Example.png *License:* Public domain *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_StepOfASpiralStair.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/20/Blender3D_StepOfASpiralStair.png *License:* CC-BY-SA-3.0 *Contributors:* Myself, Model by Tanael, with kind permission. *Original artist:* SoylentGreen
- **File:Blender3D_StrandFadeSettings.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c9/Blender3D_StrandFadeSettings.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_StrandsShaderSettings.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/77/Blender3D_StrandsShaderSettings.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_TheVeryFirstParticleSystem_Render-2.49.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/87/Blender3D_TheVeryFirstParticleSystem_Render-2.49.jpg *License:* Public domain *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_ThisCubeIsNotPossible.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6b/Blender3D_ThisCubeIsNotPossible.png *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_TiliaColorAlphaResult.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5f/Blender3D_TiliaColorAlphaResult.jpg *License:* GFDL *Contributors:* Myself. based on File:Blender3D_Lisc_lipy-Transparent.png *Original artist:* SoylentGreen
- **File:Blender3D_TransformationConstraintAnim.gif** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5d/Blender3D_TransformationConstraintAnim.gif *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_TranslucencyMapSettings-2.49.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e2/Blender3D_TranslucencyMapSettings-2.49.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_TranslucencyMapTexture-2.49.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/3e/Blender3D_TranslucencyMapTexture-2.49.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_TuerklinkeTutorial-Part1-2.48a.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8f/Blender3D_TuerklinkeTutorial-Part1-2.48a.png *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_Tutorials-Textures-Bumpmap.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9f/Blender3D_Tutorials-Textures-Bumpmap.jpg *License:* GFDL *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_Tutorials-Textures-ColorBumpmap.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/35/Blender3D_Tutorials-Textures-ColorBumpmap.jpg *License:* GFDL *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_Tutorials-Textures-Colormap.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/35/Blender3D_Tutorials-Textures-Colormap.jpg *License:* GFDL *Contributors:* Myself. Moved from de.wikibooks.org *Original artist:* SoylentGreen
- **File:Blender3D_Tutorials-Textures-DiffuseMap0.2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/87/Blender3D_Tutorials-Textures-DiffuseMap0.2.jpg *License:* GFDL *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_Tutorials-Textures-DiffuseMap1.0.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f7/Blender3D_Tutorials-Textures-DiffuseMap1.0.jpg *License:* GFDL *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_Tutorials-Textures-TranslucencyAlphamap.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c4/Blender3D_Tutorials-Textures-TranslucencyAlphamap.jpg *License:* GFDL *Contributors:* Own work *Original artist:* SoylentGreen using a file from the releasenotes of Blender version 2.32
- **File:Blender3D_Tutorials-Textures-noTranslucencyAlphamap.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/85/Blender3D_Tutorials-Textures-noTranslucencyAlphamap.jpg *License:* GFDL *Contributors:* Own work *Original artist:* SoylentGreen using a file from the releasenotes of Blender version 2.32
- **File:Blender3D_UVTexforBillboards-2.49.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/79/Blender3D_UVTexforBillboards-2.49.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_UVTexforBillboardsMapInput-2.49.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/cf/Blender3D_UVTexforBillboardsMapInput-2.49.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_UVTexforBillboardsMaterialAlpha-2.49.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a5/Blender3D_UVTexforBillboardsMaterialAlpha-2.49.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_UVTexforBillboardsTextureAlpha-2.49.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/20/Blender3D_UVTexforBillboardsTextureAlpha-2.49.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_UVTexforBillboardsTime-Index-2.49.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/62/Blender3D_UVTexforBillboardsTime-Index-2.49.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:Blender3D_ViewCoordinatesProjectionPlane.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0b/Blender3D_ViewCoordinatesProjectionPlane.jpg *License:* CC-BY-SA-3.0 *Contributors:* Own work *Original artist:* SoylentGreen
- **File:Blender3D_WIKI-HEAD.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/53/Blender3D_WIKI-HEAD.jpg *License:* CC-BY-SA-3.0 *Contributors:* ? *Original artist:* ?

- **File:Blender3D_WaveModifierParameters.png** Source: https://upload.wikimedia.org/wikipedia/commons/c/c9/Blender3D_WaveModifierParameters.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_Weight_Spec.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/b/b5/Blender3D_Weight_Spec.jpg License: Copyrighted free use Contributors: [1]. Original artist: Gabio
- **File:Blender3D_WhatIsA3DSpace.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/7/7f/Blender3D_WhatIsA3DSpace.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_ausschnitt_sw_muster.png** Source: https://upload.wikimedia.org/wikipedia/commons/d/da/Blender3D_ausschnitt_sw_muster.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3D_bevel_ausfuehrung.png** Source: https://upload.wikimedia.org/wikipedia/commons/f/f6/Blender3D_bevel_ausfuehrung.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3D_com_key_chroma.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/3/3a/Blender3D_com_key_chroma.jpg License: CC BY 2.5 Contributors: Myself, based on files from the film Elephants Dream, made by the http://www.blender.org Blender Foundation Original artist: BenutzernameAufCommons
- **File:Blender3D_draw_Vnormals.png** Source: https://upload.wikimedia.org/wikipedia/commons/f/f5/Blender3D_draw_Vnormals.png License: CC-BY-SA-3.0 Contributors: Myself Original artist: Toni Grappa
- **File:Blender3D_draw_bevel_weight.png** Source: https://upload.wikimedia.org/wikipedia/commons/d/df/Blender3D_draw_bevel_weight.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: Toni Grappa
- **File:Blender3D_draw_creases.png** Source: https://upload.wikimedia.org/wikipedia/commons/e/e4/Blender3D_draw_creases.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: Toni Grappa
- **File:Blender3D_draw_edge_angels.png** Source: https://upload.wikimedia.org/wikipedia/commons/a/a8/Blender3D_draw_edge_angels.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: Toni Grappa
- **File:Blender3D_edge_slide.png** Source: https://upload.wikimedia.org/wikipedia/commons/0/0f/Blender3D_edge_slide.png License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3D_kachel_color_picker.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/1/15/Blender3D_kachel_color_picker.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: Toni Grappa
- **File:Blender3D_kolben_zylinder_relations.png** Source: https://upload.wikimedia.org/wikipedia/commons/7/79/Blender3D_kolben_zylinder_relations.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: Soylentgreen
- **File:Blender3D_li_schreibtischlampe_screenshot.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/d/d4/Blender3D_li_schreibtischlampe_screenshot.jpg License: GPL Contributors: Own work Original artist: Toni Grappa; Veröffentlichung mit freundlicher Genehmigung von Paddy - blend.polis
- **File:Blender3D_nod_com_color_spill.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/b/bd/Blender3D_nod_com_color_spill.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3D_nod_com_flip.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/0/06/Blender3D_nod_com_flip.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3D_nod_com_glare.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/f/f8/Blender3D_nod_com_glare.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3D_nod_com_mapping_kacheln.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/f/f8/Blender3D_nod_com_mapping_kacheln.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3D_nod_com_scale.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/2/24/Blender3D_nod_com_scale.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3D_sky_menuue.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/b/b8/Blender3D_sky_menuue.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3D_tastatur.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/7/70/Blender3D_tastatur.jpg License: CC-BY-SA-3.0 Contributors: Myself Original artist: Toni Grappa
- **File:Blender3D_tut_vid_mesh.png** Source: https://upload.wikimedia.org/wikipedia/commons/7/77/Blender3D_tut_vid_mesh.png License: CC-BY-SA-3.0 Contributors: Own work Original artist: Toni Grappa
- **File:Blender3D_uv_composite_preview.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/b/b7/Blender3D_uv_composite_preview.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: Toni Grappa
- **File:Blender3D_uv_menueleiste.png** Source: https://upload.wikimedia.org/wikipedia/commons/e/e8/Blender3D_uv_menueleiste.png License: GPL Contributors: Own work Original artist: Toni Grappa
- **File:Blender3D_uv_paint_menuue_a.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/a/aa/Blender3D_uv_paint_menuue_a.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3D_uv_paint_pinsel.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/5/5e/Blender3D_uv_paint_pinsel.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3Dview.jpg** Source: <https://upload.wikimedia.org/wikipedia/commons/5/5e/Blender3Dview.jpg> License: CC-BY-SA-3.0 Contributors: Transferred from fr.wikiversity to Commons. Original artist: The original uploader was MikaYuoadas at French Wikiversity
- **File:Blender3d-ejemplo_curvas_bezier_puntoC.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/d/d4/Blender3d-ejemplo_curvas_bezier_puntoC.jpg License: CC-BY-SA-3.0 Contributors: Wikibooks. Original artist: Unknown

- **File:Blender3d-ejemplo_logo_2d.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9e/Blender3d-ejemplo_logo_2d.jpg *License:* CC-BY-SA-3.0 *Contributors:* Wikibooks. *Original artist:* ?
- **File:Blender3d-heightmap-1.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/ea/Blender3d-heightmap-1.jpg> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Ascen at English Wikibooks
- **File:Blender3d-heightmap-2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/55/Blender3d-heightmap-2.png> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Ascen at English Wikibooks
- **File:Blender3d_2d_Logo_center_3d_lifted.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0c/Blender3d_2d_Logo_center_3d_lifted.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Add_meta_ball.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fe/Blender3d_Add_meta_ball.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_House_extrude_top_of_fence_post.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d0/Blender3d_House_extrude_top_of_fence_post.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_House_fence_extrude_depth.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fb/Blender3d_House_fence_extrude_depth.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_House_scale_the_plane_to_a_fence_post.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b1/Blender3d_House_scale_the_plane_to_a_fence_post.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_House_scale_top_of_fence_post.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5d/Blender3d_House_scale_top_of_fence_post.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Jeep_bed_add_faces_a_to_h_pt1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/ac/Blender3d_Jeep_bed_add_faces_a_to_h_pt1.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Jeep_bed_add_faces_j_to_n_pt2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/86/Blender3d_Jeep_bed_add_faces_j_to_n_pt2.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Jeep_check_chasis_normals.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b1/Blender3d_Jeep_check_chasis_normals.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Jeep_doors_put_back_the_faces.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/11/Blender3d_Jeep_doors_put_back_the_faces.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Jeep_edges_to_resize_the_side.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9d/Blender3d_Jeep_edges_to_resize_the_side.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Jeep_method_2_delete_hidden_faces_v2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/de/Blender3d_Jeep_method_2_delete_hidden_faces_v2.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Jeep_method_2_extrude_up_bed_faces_v2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/aa/Blender3d_Jeep_method_2_extrude_up_bed_faces_v2.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Jeep_multicut_door_sub_method_2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/82/Blender3d_Jeep_multicut_door_sub_method_2.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Jeep_multicut_door_sub_method_2_move_down.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b8/Blender3d_Jeep_multicut_door_sub_method_2_move_down.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Jeep_resizing_the_windshield.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8b/Blender3d_Jeep_resizing_the_windshield.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Jeep_scaling_the_seat_back.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f8/Blender3d_Jeep_scaling_the_seat_back.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Jeep_select_to_subdivide_doors.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b7/Blender3d_Jeep_select_to_subdivide_doors.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Logo_combine_into_one_unit.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/44/Blender3d_Logo_combine_into_one_unit.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Meta_ball_man.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d0/Blender3d_Meta_ball_man.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Meta_ball_man_meta_mess.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/75/Blender3d_Meta_ball_man_meta_mess.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Method_2_delete_door_edge_s4.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/92/Blender3d_Method_2_delete_door_edge_s4.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Method_2_extrude_down_bed_and_doors_s2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fd/Blender3d_Method_2_extrude_down_bed_and_doors_s2.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Method_2_extrude_up_bed_faces_s6.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6b/Blender3d_Method_2_extrude_up_bed_faces_s6.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_Method_2_extrude_up_door_faces_s5.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/93/Blender3d_Method_2_extrude_up_door_faces_s5.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts

- **File:Blender3d_Method_2_select_bed_and_doors_s1.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/4/4b/Blender3d_Method_2_select_bed_and_doors_s1.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: pearts
- **File:Blender3d_Method_2_select_door_edge_s3.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/2/22/Blender3d_Method_2_select_door_edge_s3.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: pearts
- **File:Blender3d_NormalKoordinates.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/e/e2/Blender3d_NormalKoordinates.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3d_Parachute_move_center_vertex_up.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/4/45/Blender3d_Parachute_move_center_vertex_up.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: pearts
- **File:Blender3d_Parachute_select_outer_edges.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/e/e0/Blender3d_Parachute_select_outer_edges.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: pearts
- **File:Blender3d_ParentChildCoordinateAnimation.gif** Source: https://upload.wikimedia.org/wikipedia/commons/a/ac/Blender3d_ParentChildCoordinateAnimation.gif License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3d_RL_left_side_where_to_creatse.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/7/73/Blender3d_RL_left_side_where_to_creatse.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: pearts
- **File:Blender3d_RL_rocket_where_to_creatse.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/8/8f/Blender3d_RL_rocket_where_to_creatse.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: pearts
- **File:Blender3d_Rocket_launcher_right_end_vertices.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/0/0f/Blender3d_Rocket_launcher_right_end_vertices.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: pearts
- **File:Blender3d_Silver_goblet_e2_scaling.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/5/5c/Blender3d_Silver_goblet_e2_scaling.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: pearts
- **File:Blender3d_Silver_goblet_extrusion_plan.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/e/e3/Blender3d_Silver_goblet_extrusion_plan.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: pearts
- **File:Blender3d_ViewCoordinates.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/6/6c/Blender3d_ViewCoordinates.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: SoylentGreen
- **File:Blender3d_bezier_Curve_and_surface_panel.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/c/cf/Blender3d_bezier_Curve_and_surface_panel.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: pearts
- **File:Blender3d_draw_bounding_box.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/e/e3/Blender3d_draw_bounding_box.jpg License: GPL Contributors: Screenshot of the GPL program BlenderScreenshot of the GPL program Blender original blendfile and permission from Endi -Endre Barath-[1] Original artist: User:Toni Grappa
- **File:Blender3d_draw_wire.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/c/ca/Blender3d_draw_wire.jpg License: GPL Contributors: Screenshot of the GPL program Blender original blendfile and permission from Endi -Endre Barath-[1] Original artist: User:Toni Grappa
- **File:Blender3d_emulate_numpad.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/3/3a/Blender3d_emulate_numpad.jpg License: Public domain Contributors: http://en.wikibooks.org/wiki/File:Blender3d_emulate_numpad.jpg Original artist: b:en:User:Lazy-lump
- **File:Blender3d_extrude_and_scale_rocket.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/9/9e/Blender3d_extrude_and_scale_rocket.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: pearts
- **File:Blender3d_jeep_method_2_add_bottom_faces.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/f/fb/Blender3d_jeep_method_2_add_bottom_faces.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: pearts
- **File:Blender3d_nod_com_HSV.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/0/0c/Blender3d_nod_com_HSV.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3d_nod_com_alpha_over.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/4/47/Blender3d_nod_com_alpha_over.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3d_nod_com_con_color_ramp.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/f/fd/Blender3d_nod_com_con_color_ramp.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3d_nod_com_con_combine_YUFA.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/7/73/Blender3d_nod_com_con_combine_YUFA.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3d_nod_com_difference_key.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/9/9c/Blender3d_nod_com_difference_key.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3d_nod_com_image_generated.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/3/35/Blender3d_nod_com_image_generated.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3d_nod_com_math.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/e/ef/Blender3d_nod_com_math.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3d_nod_com_math_beispiel.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/6/63/Blender3d_nod_com_math_beispiel.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3d_nod_com_normalize.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/9/91/Blender3d_nod_com_normalize.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3d_nod_com_rotate.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/c/ca/Blender3d_nod_com_rotate.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa
- **File:Blender3d_nod_com_texture_value.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/c/c6/Blender3d_nod_com_texture_value.jpg License: GPL Contributors: Screenshot of the GPL program Blender Original artist: User:Toni Grappa

- **File:Blender3d_nod_com_tonemap.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/60/Blender3d_nod_com_tonemap.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* User:Toni Grappa
- **File:Blender3d_simple_house_final_render.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c7/Blender3d_simple_house_final_render.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender3d_view_name_setting_location.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9c/Blender3d_view_name_setting_location.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:BlenderAddSubsurf.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e1/BlenderAddSubsurf.jpg> *License:* GPL *Contributors:* Blender 2.4 *Original artist:* Blender Foundation
- **File:BlenderButtonsScene.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9c/BlenderButtonsScene.jpg> *License:* GPL *Contributors:* Originally from fr.wikiversity; description page is/was here. *Original artist:* Original uploader was MikaYuoadas at fr.wikiversity
- **File:BlenderCommonButtonswindow.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/7d/BlenderCommonButtonswindow.png> *License:* GPL *Contributors:* Own work (Original text: *self-made*) *Original artist:* Boyage
- **File:BlenderCommonEditing-on.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/4a/BlenderCommonEditing-on.png> *License:* GPL *Contributors:* Own work (Original text: *self-made*) *Original artist:* Boyage
- **File:BlenderCommonLSTV-on.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c1/BlenderCommonLSTV-on.png> *License:* GPL *Contributors:* Own work (Original text: *self-made*) *Original artist:* Boyage
- **File:BlenderCompositingPortalComposite.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/60/BlenderCompositingPortalComposite.png> *License:* GPL *Contributors:* Blender 2.70 *Original artist:* ?
- **File:BlenderCompositiongPortalScene1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/96/BlenderCompositiongPortalScene1.png> *License:* GPL *Contributors:* Blender 2.70 *Original artist:* ?
- **File:BlenderCompositiongPortalScene2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2c/BlenderCompositiongPortalScene2.png> *License:* GPL *Contributors:* Blender 2.70 *Original artist:* ?
- **File:BlenderCompositiongPortalScene3.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/01/BlenderCompositiongPortalScene3.png> *License:* GPL *Contributors:* Blender 2.70 *Original artist:* ?
- **File:BlenderCoordinates.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b3/BlenderCoordinates.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:BlenderCtrlLmbFace.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a7/BlenderCtrlLmbFace.png> *License:* GPL *Contributors:* Blender 2.63 *Original artist:* ?
- **File:BlenderCubeEdit.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/29/BlenderCubeEdit.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was SchmS at English Wikibooks
- **File:BlenderCubeSubsurfed.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/22/BlenderCubeSubsurfed.jpg> *License:* GPL *Contributors:* Blender 2.4 *Original artist:* Blender Foundation
- **File:BlenderCureur3D.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/60/BlenderCureur3D.jpg> *License:* GPL *Contributors:* Transferred from fr.wikiversity to Commons. *Original artist:* The original uploader was MikaYuoadas at French Wikiversity
- **File:BlenderCyclesPolarToRectangular.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/bd/BlenderCyclesPolarToRectangular.png> *License:* GPL *Contributors:* Blender 2.70 *Original artist:* ?
- **File:BlenderCyclesProceduralEyeballInner1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/53/BlenderCyclesProceduralEyeballInner1.png> *License:* GPL *Contributors:* Blender 2.70 *Original artist:* ?
- **File:BlenderCyclesProceduralEyeballInner2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/05/BlenderCyclesProceduralEyeballInner2.png> *License:* GPL *Contributors:* Blender 2.70 *Original artist:* ?
- **File:BlenderCyclesProceduralEyeballInner3.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/10/BlenderCyclesProceduralEyeballInner3.png> *License:* GPL *Contributors:* Blender 2.70 *Original artist:* ?
- **File:BlenderCyclesProceduralEyeballIrisNodes1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/ee/BlenderCyclesProceduralEyeballIrisNodes1.png> *License:* GPL *Contributors:* Blender 2.70 *Original artist:* ?
- **File:BlenderCyclesProceduralEyeballIrisNodes2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c3/BlenderCyclesProceduralEyeballIrisNodes2.png> *License:* GPL *Contributors:* Blender 2.70 *Original artist:* ?
- **File:BlenderCyclesProceduralEyeballIrisUV.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d9/BlenderCyclesProceduralEyeballIrisUV.png> *License:* GPL *Contributors:* Blender 2.70 *Original artist:* ?
- **File:BlenderCyclesProceduralEyeballResult.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/30/BlenderCyclesProceduralEyeballResult.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:BlenderCyclesProceduralEyeballScleraNodes1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/8f/BlenderCyclesProceduralEyeballScleraNodes1.png> *License:* GPL *Contributors:* Blender 2.70 *Original artist:* ?
- **File:BlenderCyclesRescale.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f2/BlenderCyclesRescale.png> *License:* GPL *Contributors:* Blender 2.70 *Original artist:* ?
- **File:BlenderD_CuttingThroughSteel-AddScene-2.48a.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6d/BlenderD_CuttingThroughSteel-AddScene-2.48a.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* SoylentGreen
- **File:BlenderDegr.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/0c/BlenderDegr.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:BlenderDieanotherway3-00.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/94/BlenderDieanotherway3-00.png> *License:* CC-BY-SA-3.0 *Contributors:* Own work (Original text: *self-made*) *Original artist:* Boyage

- **File:BlenderDrawFaceNormals.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/86/BlenderDrawFaceNormals.png> *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* AlphaEndgame
- **File:BlenderExamplePopupError.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/25/BlenderExamplePopupError.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* Lawrence D'Oliveiro
- **File:BlenderExtrudedLeg.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/be/BlenderExtrudedLeg.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was SchmS at English Wikibooks
- **File:BlenderGE_StrgA.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/76/BlenderGE_StrgA.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* SaphireS
- **File:BlenderGE_T1.3.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c5/BlenderGE_T1.3.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* SaphireS
- **File:BlenderGE_T1.5.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a3/BlenderGE_T1.5.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* SaphireS
- **File:BlenderGE_T1.6.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/96/BlenderGE_T1.6.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* SaphireS
- **File:BlenderGingerBreadManExtrudedCube3D.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c5/BlenderGingerBreadManExtrudedCube3D.png> *License:* CC BY-SA 2.5 *Contributors:* Originally from en.wikibooks; description page is/was here. *Original artist:* SatanClaus at English Wikibooks
- **File:BlenderGingerBreadManOneLegGinger3D.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/eb/BlenderGingerBreadManOneLegGinger3D.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was SatanClaus at English Wikibooks Later versions were uploaded by Argento at en.wikibooks.
- **File:BlenderGingerRender.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/8d/BlenderGingerRender.png> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Argento at English Wikibooks
- **File:BlenderGingerTexturedRender.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5a/BlenderGingerTexturedRender.png> *License:* Public domain *Contributors:* Own work *Original artist:* DDD3x
- **File:BlenderGingerbreadManLeftLeg01.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f4/BlenderGingerbreadManLeftLeg01.png> *License:* CC BY-SA 2.5 *Contributors:* Own work *Original artist:* N comme Nul at English Wikibooks
- **File:BlenderGingerbreadManLeftLeg02.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ab/BlenderGingerbreadManLeftLeg02.png> *License:* CC BY-SA 2.5 *Contributors:* Originally from en.wikibooks; description page is/was here. *Original artist:* N comme Nul at English Wikibooks
- **File:BlenderLogicActor_Dyn.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/64/BlenderLogicActor_Dyn.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was LokiClock at English Wikibooks
- **File:BlenderLogic_ForRgt_Optm.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/4f/BlenderLogic_ForRgt_Optm.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was LokiClock at English Wikibooks
- **File:BlenderMakePyramid.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/0c/BlenderMakePyramid.png> *License:* CC0 *Contributors:* Blender 2.49B *Original artist:* Lawrence D'Oliveiro
- **File:BlenderNoobProPlatformerSetUp.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/00/BlenderNoobProPlatformerSetUp.jpg> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was LokiClock at en.wikibooks
- **File:BlenderPanoramaF.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/90/BlenderPanoramaF.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Allefant at English Wikibooks
- **File:BlenderParameterShape.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/3f/BlenderParameterShape.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:BlenderPerspNarrow.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/04/BlenderPerspNarrow.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:BlenderPerspWidepng.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9a/BlenderPerspWidepng.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:BlenderPlatonicSolids.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/08/BlenderPlatonicSolids.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:BlenderSelectionModesInSolidMode.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/eb/BlenderSelectionModesInSolidMode.png> *License:* CC BY-SA 2.5 *Contributors:* Originally from en.wikibooks; description page is/was here. *Original artist:* SatanClaus at English Wikibooks
- **File:BlenderSimpleSkirt.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/08/BlenderSimpleSkirt.png> *License:* CC BY 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:BlenderSpintoolIntroDegrParameter245degrees.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c9/BlenderSpintoolIntroDegrParameter245degrees.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:BlenderSpintoolIntroDegrParameter360degrees.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/35/BlenderSpintoolIntroDegrParameter360degrees.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* <Blender Foundation

- **File:BlenderSpintoolIntroStepsParameter32Steps.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/df/BlenderSpintoolIntroStepsParameter32Steps.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:BlenderSpintoolIntroStepsParameter4Steps.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/66/BlenderSpintoolIntroStepsParameter4Steps.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:BlenderSteps.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/da/BlenderSteps.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:BlenderSubdividedCube.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2f/BlenderSubdividedCube.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was SchmS at English Wikibooks
- **File:BlenderSubsurfModifier.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/56/BlenderSubsurfModifier.jpg> *License:* GPL *Contributors:* Blender 2.4 *Original artist:* Blender Foundation
- **File:BlenderSuzanneDefaultLighting.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/75/BlenderSuzanneDefaultLighting.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:BlenderSuzanneOnePointLighting.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ad/BlenderSuzanneOnePointLighting.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:BlenderSuzanneThreePointLighting.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f9/BlenderSuzanneThreePointLighting.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:BlenderSuzanneTwoPointLighting.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2e/BlenderSuzanneTwoPointLighting.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:BlenderTetrahedronAddonEnable.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e3/BlenderTetrahedronAddonEnable.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* Lawrence D'Oliveiro
- **File:BlenderTetrahedronAddonOperator.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9e/BlenderTetrahedronAddonOperator.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* Lawrence D'Oliveiro
- **File:BlenderTetrahedronAddonPanel.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a2/BlenderTetrahedronAddonPanel.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* Lawrence D'Oliveiro
- **File:BlenderTetrahedronAddonPanel2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e1/BlenderTetrahedronAddonPanel2.png> *License:* CC0 *Contributors:* Blender 2.55 *Original artist:* Lawrence D'Oliveiro
- **File:BlenderTwoLegsGingerPulled3D.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/76/BlenderTwoLegsGingerPulled3D.png> *License:* CC BY-SA 2.5 *Contributors:* Originally from en.wikibooks; description page is/was here. *Original artist:* SatanClaus at English Wikibooks Later versions were uploaded by Argento at en.wikibooks.
- **File:BlenderWikiBookCover.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/37/BlenderWikiBookCover.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Invoker
- **File:Blender_-_Penguins_to_spheres_-_feet.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/16/Blender_-_Penguins_to_spheres_-_feet.png *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Romain Behar. Original uploader was Romainbehar at en.wikibooks
- **File:Blender_-_Penguins_to_spheres_-_feet_selection.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/71/Blender_-_Penguins_to_spheres_-_feet_selection.png *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Romain Behar. Original uploader was Romainbehar at en.wikibooks
- **File:Blender_-_Penguins_to_spheres_-_feet_start.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f1/Blender_-_Penguins_to_spheres_-_feet_start.png *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Romain Behar. Original uploader was Romainbehar at en.wikibooks
- **File:Blender_-_Penguins_to_spheres_-_subsurf.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/1a/Blender_-_Penguins_to_spheres_-_subsurf.png *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Romain Behar. Original uploader was Romainbehar at en.wikibooks
- **File:Blender_-_Penguins_to_spheres_-_tail.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/84/Blender_-_Penguins_to_spheres_-_tail.png *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Romain Behar. Original uploader was Romainbehar at en.wikibooks
- **File:Blender_-_Penguins_to_spheres_-_wing_complete.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/4d/Blender_-_Penguins_to_spheres_-_wing_complete.png *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Romain Behar. Original uploader was Romainbehar at en.wikibooks
- **File:Blender_-_Penguino_spheres_-_wing_smoothing.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b0/Blender_-_Penguino_spheres_-_wing_smoothing.png *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Romain Behar. Original uploader was Romainbehar at en.wikibooks
- **File:Blender_2.45_screenshot.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8c/Blender_2.45_screenshot.jpg *License:* GPL *Contributors:* ? *Original artist:* ?
- **File:Blender_2.49b_-_house2_rendered.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f0/Blender_2.49b_-_house2_rendered.png *License:* GPL *Contributors:* self-made model *Original artist:* Blender Foundation
- **File:Blender_2.49b_-_house2_step_10.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d6/Blender_2.49b_-_house2_step_10.png *License:* GPL *Contributors:* self-made model *Original artist:* Blender Foundation

- **File:Blender_2.49b_-_house2_step_11.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ee/Blender_2.49b_-_house2_step_11.png *License:* GPL *Contributors:* self-made model *Original artist:* Blender Foundation
- **File:Blender_2.49b_-_house2_step_12.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9c/Blender_2.49b_-_house2_step_12.png *License:* GPL *Contributors:* self-made model *Original artist:* Blender Foundation
- **File:Blender_2.49b_-_house2_step_3.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/51/Blender_2.49b_-_house2_step_3.png *License:* GPL *Contributors:* self-made model *Original artist:* Blender Foundation
- **File:Blender_2.49b_-_house2_step_5.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/7d/Blender_2.49b_-_house2_step_5.png *License:* GPL *Contributors:* self-made model *Original artist:* Blender Foundation
- **File:Blender_2.49b_-_house2_step_6.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e4/Blender_2.49b_-_house2_step_6.png *License:* GPL *Contributors:* self-made model *Original artist:* Blender Foundation
- **File:Blender_2.49b_-_house2_step_7.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9b/Blender_2.49b_-_house2_step_7.png *License:* GPL *Contributors:* self-made model *Original artist:* Blender Foundation
- **File:Blender_2.49b_-_house2_step_8.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/bc/Blender_2.49b_-_house2_step_8.png *License:* GPL *Contributors:* self-made model *Original artist:* Blender Foundation
- **File:Blender_2.49b_-_house2_step_9.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/77/Blender_2.49b_-_house2_step_9.png *License:* GPL *Contributors:* self-made model *Original artist:* Blender Foundation
- **File:Blender_2.49b_camera_icon.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/53/Blender_2.49b_camera_icon.png *License:* GPL *Contributors:* self *Original artist:* Blender Foundation
- **File:Blender_2.49b_cursor.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/68/Blender_2.49b_cursor.png *License:* GPL *Contributors:* self *Original artist:* Blender Foundation
- **File:Blender_2.49b_lamp_icon.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/37/Blender_2.49b_lamp_icon.png *License:* GPL *Contributors:* self *Original artist:* Blender Foundation
- **File:Blender_2.58a_-_Face_Select_Mode.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/31/Blender_2.58a_-_Face_Select_Mode.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* bamacsguy
- **File:Blender_2.58a_-_House_Step_5_-_Pivot_Menu.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fb/Blender_2.58a_-_House_Step_5_-_Pivot_Menu.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* bamacsguy
- **File:Blender_2.59_3d_view_icon.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/dd/Blender_2.59_3d_view_icon.png *License:* GPL *Contributors:* self *Original artist:* Blender Foundation
- **File:Blender_2.59_Info_ico.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/01/Blender_2.59_Info_ico.png *License:* GPL *Contributors:* self *Original artist:* Blender Foundation
- **File:Blender_2.59_Outliner_ico.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/da/Blender_2.59_Outliner_ico.png *License:* GPL *Contributors:* self *Original artist:* Blender Foundation
- **File:Blender_2.59_Properties_ico.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fe/Blender_2.59_Properties_ico.png *License:* GPL *Contributors:* self *Original artist:* Blender Foundation
- **File:Blender_2.59_User_Prefs_ico.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0e/Blender_2.59_User_Prefs_ico.png *License:* GPL *Contributors:* self *Original artist:* Blender Foundation
- **File:Blender_2.63_init_factory_bottom.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/36/Blender_2.63_init_factory_bottom.png *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:Blender_2.75a_nice_round_wheel.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/62/Blender_2.75a_nice_round_wheel.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Blender_2.75a_subdivision_surface.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/3d/Blender_2.75a_subdivision_surface.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Blender_2.75a_subdivision_surface_Rendered.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/40/Blender_2.75a_subdivision_surface_Rendered.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Blender_2.75a_widen_chassis.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/96/Blender_2.75a_widen_chassis.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Blender_2.76b_Startup.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/ac/Blender_2.76b_Startup.png *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Spamm
- **File:Blender_2.78_Lattice_Object_Data_button.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/bc/Blender_2.78_Lattice_Object_Data_button.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:Blender_2.78_Mesh_Object_Data_Button.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9f/Blender_2.78_Mesh_Object_Data_Button.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:Blender_3D-cursor.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6a/Blender_3D-cursor.png *License:* GPL *Contributors:* ? *Original artist:* ?
- **File:Blender_3D_-_Logo_circle_in_3D_-_20100905.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d7/Blender_3D_-_Logo_circle_in_3D_-_20100905.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender_3D_Manipulator_screenshot.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/84/Blender_3D_Manipulator_screenshot.png *License:* GPL *Contributors:* Blender 2.67b *Original artist:* Blender Foundation
- **File:Blender_3D_Procedural_Wood_1_Colors_Window.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a2/Blender_3D_Procedural_Wood_1_Colors_Window.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Spiderworm at English Wikibooks

- **File:Blender_3D_Scene.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/32/Blender_3D_Scene.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* ErikBongers at English Wikibooks
- **File:Blender_3D_mini_axis_screenshot.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/bd/Blender_3D_mini_axis_screenshot.png *License:* GPL *Contributors:* Blender 2.67b *Original artist:* Blender Foundation
- **File:Blender_3d_RL_right_side_where_to_crearce.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/39/Blender_3d_RL_right_side_where_to_crearce.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender_3d_Show_normals_and_vnormals2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0b/Blender_3d_Show_normals_and_vnormals2.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* pearts
- **File:Blender_Actor.PNG** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/bb/Blender_Actor.PNG *License:* CC BY-SA 3.0 *Contributors:* screen image *Original artist:* me
- **File:Blender_Armature_Bone_icon_(selected).png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fe/Blender_Armature_Bone_icon_%28selected%29.png *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* TROPtastic
- **File:Blender_Armature_Data_icon.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/30/Blender_Armature_Data_icon.png *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* TROPtastic
- **File:Blender_Armature_Object_icon.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/cf/Blender_Armature_Object_icon.png *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* TROPtastic
- **File:Blender_Blue_Marble.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/78/Blender_Blue_Marble.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was ZeroOne at English Wikibooks
- **File:Blender_Button_arraignment.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/22/Blender_Button_arraignment.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Jeremy B
- **File:Blender_Completed_Parachute.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/48/Blender_Completed_Parachute.jpg *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Radialronnie at en.wikibooks
- **File:Blender.Controllers.PNG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/20/Blender.Controllers.PNG> *License:* CC-BY-SA-3.0 *Contributors:* screen pic *Original artist:* me
- **File:Blender_Cylinder_Taper_For_Parachute.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/2f/Blender_Cylinder_Taper_For_Parachute.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* Radialronnie
- **File:Blender_Dent.PNG** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ef/Blender_Dent.PNG *License:* CC BY-SA 3.0 *Contributors:* i mad this *Original artist:* me
- **File:Blender_Graphic_Menu.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/90/Blender_Graphic_Menu.jpg *License:* CC BY-SA 3.0 *Contributors:* Blender 3D snapshot *Original artist:* Jeremy B.
- **File:Blender_Grass.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/eb/Blender_Grass.jpg *License:* Public domain *Contributors:* Own work *Original artist:* Binky1206
- **File:Blender_HDR_AngMap.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/bf/Blender_HDR_AngMap.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* ErikBongers at English Wikibooks
- **File:Blender_HDR_Mirror.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/79/Blender_HDR_Mirror.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* ErikBongers at English Wikibooks
- **File:Blender_HDR_Result.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/10/Blender_HDR_Result.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* ErikBongers at English Wikibooks
- **File:Blender_HDR_YafRay_Result.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/99/Blender_HDR_YafRay_Result.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* ErikBongers at English Wikibooks
- **File:Blender_Handicon.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/92/Blender_Handicon.png *License:* GPL *Contributors:* ? *Original artist:* ?
- **File:Blender_Hill.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8a/Blender_Hill.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Ask-enwikibooks at English Wikibooks
- **File:Blender_Limit_selection_to_visible_Button.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5c/Blender_Limit_selection_to_visible_Button.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:Blender_Linking_Menus.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0b/Blender_Linking_Menus.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Jeremy B.
- **File:Blender_Multi_Button.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/66/Blender_Multi_Button.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Jeremy B.
- **File:Blender_Pfanne_Abbildung1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/90/Blender_Pfanne_Abbildung1.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* Mangarm
- **File:Blender_Pfanne_Abbildung2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/60/Blender_Pfanne_Abbildung2.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* Mangarm
- **File:Blender_Pfanne_Abbildung6.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a9/Blender_Pfanne_Abbildung6.jpg *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* Mangarm

- **File:Blender_Pull_Corners_Down.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/4d/Blender_Pull_Corners_Down.jpg *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Radialronnie at English Wikibooks
- **File:Blender_Screen_Parachute_shaped_Mesh.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/61/Blender_Screen_Parachute_shaped_Mesh.jpg *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User: Adrignola using CommonsHelper. *Original artist:* Original uploader was Radialronnie at en.wikibooks
- **File:Blender_Select_Corners_OnCylinder.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/82/Blender_Select_Corners_OnCylinder.jpg *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Radialronnie at English Wikibooks
- **File:Blender_SimpleHumanFigure.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/83/Blender_SimpleHumanFigure.png *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:Blender_Smooth_fallof.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/38/Blender_Smooth_fallof.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Morrolan at English Wikibooks
- **File:Blender_TexturePanelScreenShot.PNG** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/7d/Blender_TexturePanelScreenShot.PNG *License:* CC BY-SA 3.0 *Contributors:* Screenshot of the GPL program [1] *Original artist:* Anonymous_man
- **File:Blender_background_image_dialog.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d5/Blender_background_image_dialog.JPG *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Yoyotweak at English Wikibooks
- **File:Blender_bangers_and_mash.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/68/Blender_bangers_and_mash.jpg *License:* Public domain *Contributors:* Own work *Original artist:* Frogulis
- **File:Blender_basic_menu.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e6/Blender_basic_menu.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Jeremy B. Original uploader was Bullercruz1 at en.wikibooks. Later version(s) were uploaded by Adrignola at en.wikibooks.
- **File:Blender_bowl.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ef/Blender_bowl.JPG *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Jeremy B
- **File:Blender_circle_size.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a6/Blender_circle_size.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Morrolan at English Wikibooks
- **File:Blender_edit_buttons.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f1/Blender_edit_buttons.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Totophe64 at English Wikibooks
- **File:Blender_env_order.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/ff/Blender_env_order.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Glome~enwikibooks at English Wikibooks
- **File:Blender_fluid-demo.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e1/Blender_fluid-demo.jpg *License:* Public domain *Contributors:* My own work. *Original artist:* Jackson Williams (Malaz). The original uploader was Malaz at English Wikibooks
- **File:Blender_grab_1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/1f/Blender_grab_1.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Morrolan at English Wikibooks
- **File:Blender_grab_2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b4/Blender_grab_2.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Morrolan at English Wikibooks
- **File:Blender_grab_down_circle.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0e/Blender_grab_down_circle.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Morrolan at English Wikibooks
- **File:Blender_grab_down_last.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/3d/Blender_grab_down_last.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Morrolan at English Wikibooks
- **File:Blender_grab_last.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/be/Blender_grab_last.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Morrolan at English Wikibooks
- **File:Blender_heightmap_screenshot.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/11/Blender_heightmap_screenshot.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Aidan.Sullivan at English Wikibooks
- **File:Blender_hills.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/31/Blender_hills.jpg *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Blenderman~enwikibooks at English Wikibooks
- **File:Blender_icosphere_LSCM.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/38/Blender_icosphere_LSCM.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was ZeroOne at English Wikibooks
- **File:Blender_icosphere_TexFace.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c5/Blender_icosphere_TexFace.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was ZeroOne at English Wikibooks
- **File:Blender_icosphere_and_UV_Image_Editor.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/11/Blender_icosphere_and_UV_Image_Editor.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was ZeroOne at English Wikibooks

- **File:Blender_icosphere_textured.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/1b/Blender_icosphere_textured.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was ZeroOne at English Wikibooks
- **File:Blender_image_remove.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/3a/Blender_image_remove.JPG *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Yoyotweak at English Wikibooks
- **File:Blender_image_select.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8f/Blender_image_select.JPG *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Yoyotweak at English Wikibooks
- **File:Blender_image_select_button1.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/72/Blender_image_select_button1.JPG *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Yoyotweak at English Wikibooks
- **File:Blender_image_select_button2.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/ae/Blender_image_select_button2.JPG *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Yoyotweak at English Wikibooks
- **File:Blender_image_select_textbox.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/df/Blender_image_select_textbox.JPG *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Yoyotweak at English Wikibooks
- **File:Blender_knob.PNG** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0b/Blender_knob.PNG *License:* CC BY-SA 3.0 *Contributors:* i mad this *Original artist:* me
- **File:Blender_load_image_2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/04/Blender_load_image_2.jpg *License:* GPL *Contributors:* Own work *Original artist:* John Whelan
- **File:Blender_map_input.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d1/Blender_map_input.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Argento at English Wikibooks
- **File:Blender_material_volcano.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0f/Blender_material_volcano.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Morrolan at English Wikibooks
- **File:Blender_new_image.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e7/Blender_new_image.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:Blender_pasto.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c2/Blender_pasto.jpg *License:* GFDL *Contributors:* Own work *Original artist:* Imageday
- **File:Blender_plus_norm.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/94/Blender_plus_norm.JPG *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Jeremy B.
- **File:Blender_proportional_smooth_after_rotation.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b3/Blender_proportional_smooth_after_rotation.jpg *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Hendric
- **File:Blender_proportional_smooth_before_rotation.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/20/Blender_proportional_smooth_before_rotation.jpg *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Hendric
- **File:Blender_real-time_shadow.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/46/Blender_real-time_shadow.png *License:* Public domain *Contributors:* Own work *Original artist:* Banlu Kemiyatorn
- **File:Blender_refresh.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/17/Blender_refresh.JPG *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Yoyotweak at English Wikibooks
- **File:Blender_render_texture.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c1/Blender_render_texture.jpg *License:* Public domain *Contributors:* Blender 3D Render *Original artist:* Jeremy B.. The original uploader was Bullercruz1 at English Wikibooks
- **File:Blender_simple_person.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b3/Blender_simple_person.png *License:* GPL *Contributors:* ? *Original artist:* ?
- **File:Blender_tips_cube_subsurf2d.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f6/Blender_tips_cube_subsurf2d.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was LuizAlvarez at English Wikibooks
- **File:Blender_use_background_image_button.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/51/Blender_use_background_image_button.JPG *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Yoyotweak at English Wikibooks
- **File:Blender_use_background_image_button_v2p43.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c0/Blender_use_background_image_button_v2p43.JPG *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Yoyotweak at English Wikibooks
- **File:Blender_users_count.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/56/Blender_users_count.JPG *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Yoyotweak at English Wikibooks
- **File:Blender_volcano_fractal5.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/31/Blender_volcano_fractal5.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Blender_volcano_magma.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c1/Blender_volcano_magma.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Morrolan at en.wikibooks

- **File:Blender_volcano_stucci.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b8/Blender_volcano_stucci.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Morrolan at English Wikibooks
- **File:Blender_w_cube.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/50/Blender_w_cube.jpg *License:* Public domain *Contributors:* Blender 3D snapshot *Original artist:* Jeremy B.. The original uploader was Bullercruz1 at English Wikibooks
- **File:Blender_w_cube_musgrave.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/18/Blender_w_cube_musgrave.jpg *License:* Public domain *Contributors:* Blender 3D snapshot *Original artist:* Jeremy B.. The original uploader was Bullercruz1 at English Wikibooks
- **File:Blenderceramics.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/53/Blenderceramics.jpg> *License:* CC BY 3.0 *Contributors:* Own work *Original artist:* Rosver
- **File:Blenderediting.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c4/Blenderediting.png> *License:* GPL *Contributors:* ? *Original artist:* ?
- **File:Blendermaterial.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/54/Blendermaterial.png> *License:* GPL *Contributors:* ? *Original artist:* ?
- **File:Blenderpenguin1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/47/Blenderpenguin1.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Shythinker at English Wikibooks
- **File:Blenderpenguin2a.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2b/Blenderpenguin2a.png> *License:* GPL *Contributors:* Blender 249.2 *Original artist:* Screenshot of Blender by me.
- **File:Blenderpenguin2b.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/ca/Blenderpenguin2b.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Shythinker at English Wikibooks
- **File:Blenderpenguin2c.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/55/Blenderpenguin2c.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Shythinker at English Wikibooks
- **File:Blenderpenguin3c.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/10/Blenderpenguin3c.png> *License:* GPL *Contributors:* Blender screenshot. *Original artist:* ?
- **File:Blenderpenguin3c_-_2.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/ca/Blenderpenguin3c_-_2.png *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Shythinker at English Wikibooks
- **File:Blenderpenguin4-02.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/27/Blenderpenguin4-02.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Boyage. Original uploader was Boyage at en.wikibooks
- **File:Blenderpenguin4-03.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c8/Blenderpenguin4-03.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Boyage. Original uploader was Boyage at en.wikibooks
- **File:Blenderpenguin4-04.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/ed/Blenderpenguin4-04.png> *License:* CC-BY-SA-3.0 *Contributors:* Own work (Original text: *self-made*) *Original artist:* Boyage
- **File:Blenderpenguin4b.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/99/Blenderpenguin4b.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Shythinker at English Wikibooks
- **File:Blenderpenguin5-01.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b5/Blenderpenguin5-01.png> *License:* CC-BY-SA-3.0 *Contributors:* Own work (Original text: *self-made*) *Original artist:* Boyage
- **File:Blenderpenguin5-05.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/61/Blenderpenguin5-05.png> *License:* CC-BY-SA-3.0 *Contributors:* Own work (Original text: *self-made*) *Original artist:* Boyage
- **File:Blenderpenguin5-06.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2a/Blenderpenguin5-06.png> *License:* CC-BY-SA-3.0 *Contributors:* Own work (Original text: *self-made*) *Original artist:* Boyage
- **File:Blenderpenguin5-08.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/ca/Blenderpenguin5-08.png> *License:* CC-BY-SA-3.0 *Contributors:* Own work (Original text: *self-made*) *Original artist:* Boyage
- **File:Blenderpenguin5-09.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c0/Blenderpenguin5-09.png> *License:* CC-BY-SA-3.0 *Contributors:* Own work (Original text: *self-made*) *Original artist:* Boyage
- **File:Blenderpenguin5-10.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/73/Blenderpenguin5-10.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Boyage. Original uploader was Boyage at en.wikibooks
- **File:Blenderpenguin5b.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9c/Blenderpenguin5b.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Shythinker at English Wikibooks
- **File:Blenderpenguin6b.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/dd/Blenderpenguin6b.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Shythinker at English Wikibooks
- **File:Blenderpenguin7.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/ba/Blenderpenguin7.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Shythinker at English Wikibooks

- **File:Blenderpenguin8.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/52/Blenderpenguin8.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Shythinker at English Wikibooks
- **File:Blenderradio.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/36/Blenderradio.png> *License:* GPL *Contributors:* ? *Original artist:* ?
- **File:Blenderscene.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c3/Blenderscene.png> *License:* GPL *Contributors:* ? *Original artist:* ?
- **File:Blendertexture.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9b/Blendertexture.png> *License:* GPL *Contributors:* ? *Original artist:* ?
- **File:Blue(tutorialBlender).PNG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ae/Blue%28tutorialBlender%29.PNG> *License:* CCO *Contributors:* Own work *Original artist:* Vesseshhebar
- **File:BlueMarble-2001-2002.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1c/BlueMarble-2001-2002.jpg> *License:* Public domain *Contributors:* ? *Original artist:* ?
- **File:Blueeyes.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/cf/Blueeyes.jpg> *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Chinmay gautam at English Wikibooks
- **File:Bone_ana.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/dc/Bone_ana.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Bones_Tutorial_-_Finished_Render.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5c/Bones_Tutorial_-_Finished_Render.png *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* TROPtastic
- **File:Bones_Tutorial_-_Finished_Skeleton.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6a/Bones_Tutorial_-_Finished_Skeleton.png *License:* GPL *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Bones_Tutorial_-_First_Bone.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/45/Bones_Tutorial_-_First_Bone.png *License:* GPL *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Bones_Tutorial_-_Naming_Bone.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ea/Bones_Tutorial_-_Naming_Bone.png *License:* GPL *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Bones_Tutorial_-_Setup.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/30/Bones_Tutorial_-_Setup.png *License:* GPL *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Bones_Tutorial_-_Vertex_Grouping.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/bd/Bones_Tutorial_-_Vertex_Grouping.png *License:* GPL *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Book_important2.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/91/Book_important2.svg *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* darklama
- **File:BounceballlatticeSS.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b3/BounceballlatticeSS.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:BouncebasekeysSS.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/20/BouncebasekeysSS.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:BounceipoSS.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/51/BounceipoSS.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:BounceipokeysSS.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5a/BounceipokeysSS.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:Bouncelattmod1SS.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/91/Bouncelattmod1SS.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:Bouncelattmod2SS.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9a/Bouncelattmod2SS.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:BouncepivotSS.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/91/BouncepivotSS.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:Bouncewiki.gif** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/48/Bouncewiki.gif> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* AndyD at English Wikibooks
- **File:BoxModelingIntro_AddingDetails.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/73/BoxModelingIntro_AddingDetails.jpg *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:BoxModelingIntro_Form.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/39/BoxModelingIntro_Form.jpg *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:BoxModelingIntro_Primitive.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/64/BoxModelingIntro_Primitive.jpg *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:BoxModelingSwanChairDetailing1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5f/BoxModelingSwanChairDetailing1.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver

- **File:BoxModelingSwanChairLeg24.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/5f/BoxModelingSwanChairLeg24.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairLeg25.png** Source: <https://upload.wikimedia.org/wikipedia/commons/3/33/BoxModelingSwanChairLeg25.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairLeg26.png** Source: <https://upload.wikimedia.org/wikipedia/commons/4/43/BoxModelingSwanChairLeg26.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairLeg27.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/df/BoxModelingSwanChairLeg27.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairLeg29.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f5/BoxModelingSwanChairLeg29.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairLeg3.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/60/BoxModelingSwanChairLeg3.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairLeg30.png** Source: <https://upload.wikimedia.org/wikipedia/commons/c/ca/BoxModelingSwanChairLeg30.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairLeg31.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/12/BoxModelingSwanChairLeg31.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairLeg4.5.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/b8/BoxModelingSwanChairLeg4.5.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairLeg4.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/b6/BoxModelingSwanChairLeg4.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairLeg5.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/00/BoxModelingSwanChairLeg5.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairLeg6.png** Source: <https://upload.wikimedia.org/wikipedia/commons/c/cd/BoxModelingSwanChairLeg6.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairLeg7.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/01/BoxModelingSwanChairLeg7.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairLeg8.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/74/BoxModelingSwanChairLeg8.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairLeg9.png** Source: <https://upload.wikimedia.org/wikipedia/commons/4/42/BoxModelingSwanChairLeg9.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairTutorialAddingMirrorModifier.png** Source: <https://upload.wikimedia.org/wikipedia/commons/d/d8/BoxModelingSwanChairTutorialAddingMirrorModifier.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairTutorialShapingChair1.png** Source: <https://upload.wikimedia.org/wikipedia/commons/0/03/BoxModelingSwanChairTutorialShapingChair1.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairTutorialShapingChair10.png** Source: <https://upload.wikimedia.org/wikipedia/commons/c/cf/BoxModelingSwanChairTutorialShapingChair10.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairTutorialShapingChair11.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f9/BoxModelingSwanChairTutorialShapingChair11.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairTutorialShapingChair12.png** Source: <https://upload.wikimedia.org/wikipedia/commons/3/37/BoxModelingSwanChairTutorialShapingChair12.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairTutorialShapingChair13.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/8e/BoxModelingSwanChairTutorialShapingChair13.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairTutorialShapingChair14.png** Source: <https://upload.wikimedia.org/wikipedia/commons/e/e0/BoxModelingSwanChairTutorialShapingChair14.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairTutorialShapingChair15.png** Source: <https://upload.wikimedia.org/wikipedia/commons/8/8c/BoxModelingSwanChairTutorialShapingChair15.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairTutorialShapingChair2.png** Source: <https://upload.wikimedia.org/wikipedia/commons/a/ad/BoxModelingSwanChairTutorialShapingChair2.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairTutorialShapingChair4.png** Source: <https://upload.wikimedia.org/wikipedia/commons/7/73/BoxModelingSwanChairTutorialShapingChair4.png> License: CC0 Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairTutorialShapingChair7.png** Source: <https://upload.wikimedia.org/wikipedia/commons/2/2f/BoxModelingSwanChairTutorialShapingChair7.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairTutorialShapingChair8.png** Source: <https://upload.wikimedia.org/wikipedia/commons/a/aa/BoxModelingSwanChairTutorialShapingChair8.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:BoxModelingSwanChairTutorialShapingChair9.png** Source: <https://upload.wikimedia.org/wikipedia/commons/b/b4/BoxModelingSwanChairTutorialShapingChair9.png> License: Public domain Contributors: Own work Original artist: Rosver
- **File:Boxselect.png** Source: <https://upload.wikimedia.org/wikipedia/commons/f/f0/Boxselect.png> License: CC BY-SA 3.0 Contributors: Own work Original artist: JonnyJack
- **File:BufShad.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/64/BufShad.png> License: GPL Contributors: Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. Original artist: The original uploader was Totophe64 at English Wikibooks

- **File:C_IK_panel.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ef/C_IK_panel.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Gabio at English Wikibooks
- **File:C_ik.gif** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9c/C_ik.gif *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:C_ik_tree.gif** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b9/C_ik_tree.gif *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:C_loc.gif** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ea/C_loc.gif *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Gabio at en.wikibooks
- **File:C_loc_panel.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0d/C_loc_panel.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Gabio at English Wikibooks
- **File:C_loc_track.gif** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/7e/C_loc_track.gif *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:C_rot.gif** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/70/C_rot.gif *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:C_rot_panel.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/41/C_rot_panel.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Gabio at English Wikibooks
- **File:C_track.gif** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f7/C_track.gif *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:C_track_panel.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b8/C_track_panel.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Gabio at English Wikibooks
- **File:Camouflage_Jeep_with_Rocket.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/be/Camouflage_Jeep_with_Rocket.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Carpet-Material-Final.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a3/Carpet-Material-Final.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dan13 at English Wikibooks
- **File:Cartesian.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/6c/Cartesian.png> *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Smjefferson602
- **File:CeramicMagentaBall.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/56/CeramicMagentaBall.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:ChairTutorial1.4.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f5/ChairTutorial1.4.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:ChairTutorialPreparationLoopStudy.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e5/ChairTutorialPreparationLoopStudy.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:Changewindow.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/4b/Changewindow.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Tjb0607 at English Wikibooks
- **File:Checkerboard_tile.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5f/Checkerboard_tile.svg *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:CircleSquare2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c3/CircleSquare2.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Darkonk at en.wikibooks
- **File:Clayman.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/37/Clayman.jpg> *License:* CC BY-SA 2.0 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was Bootaleg at English Wikibooks
- **File:Clear_Blender.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/90/Clear_Blender.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Jeremy B. Original uploader was Bullercruz1 at en.wikibooks. Later version(s) were uploaded by Adrignola at en.wikibooks.
- **File:Clipboard.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1f/Clipboard.svg> *License:* GPL *Contributors:* Own work, based on File:Evolution-tasks-old.png, which was released into the public domain by its creator AzaToth. *Original artist:* Tkgd2007
- **File:Cloudsettings.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1c/Cloudsettings.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* SergeantOreo at English Wikibooks
- **File:Color_wheel_numbered.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/59/Color_wheel_numbered.png *License:* CC BY-SA 1.0 *Contributors:* self *Original artist:* Blender Foundation
- **File:Commons-logo.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/4a/Commons-logo.svg> *License:* Public domain *Contributors:* This version created by Pumbaa, using a proper partial circle and SVG geometry features. (Former versions used to be slightly warped.) *Original artist:* SVG version was created by User:Grunt and cleaned up by 3247, based on the earlier PNG version, created by Reidab.
- **File:CompositeNode-Settings_withText_BlenderMistTutorial.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ed/CompositeNode-Settings_withText_BlenderMistTutorial.jpg *License:* GPL *Contributors:* Blender *Original artist:* ?

- **File:ConeCurve1.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/45/ConeCurve1.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Embri at English Wikibooks
- **File:ConeCurve2.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c2/ConeCurve2.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Embri at English Wikibooks
- **File:ConeCurve3.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/43/ConeCurve3.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Embri at English Wikibooks
- **File:ConeCurve4.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/82/ConeCurve4.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Embri
- **File:Cornea-example.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ad/Cornea-example.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was Icky-wiki at English Wikibooks
- **File:Cornea_Curve.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c3/Cornea_Curve.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:Cornea_Triangles.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8b/Cornea_Triangles.jpg *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:Coulours.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/ec/Coulours.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Totophe64 at English Wikibooks
- **File:Crater_done.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/99/Crater_done.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Crater_done_Rendered.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/eb/Crater_done_Rendered.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Creasing-five-spots-selected.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/7f/Creasing-five-spots-selected.png> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Wynnmc at en.wikibooks
- **File:CrudeCyclesRender1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/8d/CrudeCyclesRender1.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:CrudeCyclesRender2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/8e/CrudeCyclesRender2.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:CrudeCyclesRender3.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e7/CrudeCyclesRender3.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:Cube_perpective_mode.PNG** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/2a/Cube_perpective_mode.PNG *License:* CC0 *Contributors:* Own work *Original artist:* ?
- **File:CursorSelectionSnapDemoAfter.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/cf/CursorSelectionSnapDemoAfter.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:CursorSelectionSnapDemo_Before.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/36/CursorSelectionSnapDemo_Before.jpg *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:Curve_and_Surface_Box.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e3/Curve_and_Surface_Box.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Lazy-lump at English Wikibooks
- **File:CyclesGlobalIlluminationExample.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/68/CyclesGlobalIlluminationExample.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:Cylinder.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/58/Cylinder.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* JonnyJack
- **File:DOLPHINS_IN_SEA.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/29/DOLPHINS_IN_SEA.jpg *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Deleting_faces.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/17/Deleting_faces.png *License:* Public domain *Contributors:* Desktop screen capture, my PC at home *Original artist:* Paul James Chatman
- **File:DieAnotherWay12c.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/54/DieAnotherWay12c.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Shythinker at English Wikibooks
- **File:DieAnotherWay18.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/0f/DieAnotherWay18.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Shythinker at English Wikibooks
- **File:Domain-basic-fluid-tutorial-blender2.76.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/82/Domain-basic-fluid-tutorial-blender2.76.jpg> *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Dragon_003.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e2/Dragon_003.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Dragon_008.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/7e/Dragon_008.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike

- **File:Dragon_009.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f6/Dragon_009.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Dragon_013.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b7/Dragon_013.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Dragon_015.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/56/Dragon_015.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Dragon_016.png.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/36/Dragon_016.jpg *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Dragon_017.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/56/Dragon_017.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Dragon_017a.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/09/Dragon_017a.png *License:* CC BY-SA 2.0 *Contributors:* Screenshot of the GPL program Blender *Original artist:* MercedMike
- **File:Dragon_023.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9e/Dragon_023.png *License:* CC BY-SA 2.0 *Contributors:* Screenshot of the GPL program Blender *Original artist:* MercedMike
- **File:Dragon_025.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6a/Dragon_025.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Dragon_026.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/97/Dragon_026.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Dragon_027.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/28/Dragon_027.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Dragon_028.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b0/Dragon_028.jpg *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Dragon_029.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fd/Dragon_029.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Dragon_031.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6c/Dragon_031.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* MercedMike
- **File:Dragonsketch.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/84/Dragonsketch.jpg> *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* LightspeedLife
- **File:Drawbeziercurves.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e1/Drawbeziercurves.png> *License:* GPL *Contributors:* Inkscape *Original artist:* User:Tjb0607
- **File:Dropdown_-_2.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/16/Dropdown_-_2.png *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Gregggreg at English Wikibooks
- **File:DuplVerts_in_Blender_2.78_interface.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/90/DuplVerts_in_Blender_2.78_interface.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* Chloris Pale Green
- **File:Duplicate_Material(tutorialBlender).png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5b/Duplicate_Material%28tutorialBlender%29.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Vessesh Hebbar
- **File>Edit_panel_bones.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/dc/Edit_panel_bones.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Gabio at en.wikibooks
- **File:Editing-panel-modifiers-subsurf.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/bf/Editing-panel-modifiers-subsurf.png> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Wynnmc at en.wikibooks
- **File:Editing.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/37/Editing.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Gregggreg at English Wikibooks
- **File:Editmodewithhead.PNG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f4/Editmodewithhead.PNG> *License:* Public domain *Contributors:* "Screenshot from Blender3D softwared (GPL) by Jake.lewis4" *Original artist:* Jake.lewis4
- **File:Eighth-cut-removed.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/56/Eighth-cut-removed.png> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Wynnmc at en.wikibooks
- **File:EnableDupVerts.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d2/EnableDupVerts.png> *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Radialronnie at English Wikibooks
- **File:Env_bone.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8a/Env_bone.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Gabio at en.wikibooks
- **File:Example_normal_color_map.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/da/Example_normal_color_map.JPG *License:* CC BY-SA 3.0 *Contributors:* Blender 3D Render *Original artist:* Jeremy B.
- **File:Example_normal_texture_1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9c/Example_normal_texture_1.jpg *License:* CC BY-SA 3.0 *Contributors:* Blender 3D Render *Original artist:* Jeremy B.
- **File:Example_normal_texture_2.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b1/Example_normal_texture_2.JPG *License:* CC BY-SA 3.0 *Contributors:* Blender 3D Render *Original artist:* Jeremy B.

- **File:Extra_examples.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/ac/Extra_examples.jpg *License:* Public domain *Contributors:* Multiple Blender 3D Renders spliced together *Original artist:* Jeremy B.. The original uploader was Bullercruz1 at English Wikibooks
- **File:Extrudeandscale.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a7/Extrudeandscale.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* JonnyJack
- **File:ExtrudedCube1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/ff/ExtrudedCube1.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Argento at English Wikibooks
- **File:ExtrudedCube2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/6e/ExtrudedCube2.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Argento at English Wikibooks
- **File:Eyeball_texture.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f0/Eyeball_texture.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Face_Bezier_Curve.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e3/Face_Bezier_Curve.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Lazy-lump at English Wikibooks
- **File:Falloff_Edit.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/91/Falloff_Edit.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:Featured_book_en.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ee/Featured_book_en.svg *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Darklama
- **File:Fence-complete.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/7e/Fence-complete.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Wynnmc at English Wikibooks
- **File:Fifth-cut-removed.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ad/Fifth-cut-removed.png> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Wynnmc at en.wikibooks
- **File:Fifth-cut-selected.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1a/Fifth-cut-selected.png> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Wynnmc at en.wikibooks
- **File:File-BoxModelingSwanChairTutorialShapingChair5.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e7/File-BoxModelingSwanChairTutorialShapingChair5.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:FinalTire.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/3f/FinalTire.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:First_angle_projecting.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fb/First_angle_projecting.png *License:* CC-BY-SA-3.0 *Contributors:* ? *Original artist:* ?
- **File:First_key_1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e1/First_key_1.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Gabio at English Wikibooks
- **File:First_key_2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/36/First_key_2.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Gabio at English Wikibooks
- **File:Firstcircle.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/27/Firstcircle.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Darkonc at English Wikibooks
- **File:Flickr_2.0_lic_Komodo_Dragon_SKIN.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0b/Flickr_2.0_lic_Komodo_Dragon_SKIN.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Flickr_2.0_lic_flames.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/74/Flickr_2.0_lic_flames.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Flickr_textures_brick_wall.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/1c/Flickr_textures_brick_wall.jpg *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Fluid-basic-fluid-tutorial-blender2.76.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/ea/Fluid-basic-fluid-tutorial-blender2.76.jpg> *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Front-face-9v-cuts.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e1/Front-face-9v-cuts.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Wynnmc at English Wikibooks
- **File:Fur_settings.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a3/Fur_settings.png *License:* CC BY-SA 3.0 *Contributors:* Screenshot of the GPL program Blender *Original artist:* MercedMike
- **File:Generi_RVK.gif** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a0/Generi_RVK.gif *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Getting_Started_With_Bob.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/56/Getting_Started_With_Bob.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Wavez at English Wikibooks
- **File:Ginger-smoothall.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f6/Ginger-smoothall.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Darkonc at English Wikibooks

- **File:Ginger-smoothbody.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/41/Ginger-smoothbody.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Darkonc at English Wikibooks
- **File:Ginger-zscale.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/0b/Ginger-zscale.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Darkonc at English Wikibooks
- **File:Glasses_800_edit.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ec/Glasses_800_edit.png *License:* Public domain *Contributors:* <http://www.oyonale.com/modeles.php?lang=en&page=40> *Original artist:* Gilles Tran
- **File:GlowingCubeWithEdgeOutline.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/aa/GlowingCubeWithEdgeOutline.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:GlowingCubeWithEdgeOutlineBlurred.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/42/GlowingCubeWithEdgeOutlineBlurred.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:Gnome-mime-application-pdf.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b6/Gnome-mime-application-pdf.svg> *License:* LGPL *Contributors:* <http://ftp.gnome.org/pub/GNOME/sources/gnome-themes-extras/0.9/gnome-themes-extras-0.9.0.tar.gz> *Original artist:* David Vignoni
- **File:Goblet-lettered-lines.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/46/Goblet-lettered-lines.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Wynnmc at English Wikibooks
- **File:Goblet-smoothed-sharpened-front.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/97/Goblet-smoothed-sharpened-front.png> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Wynnmc at en.wikibooks
- **File:Goblet-smoothed-sharpened-top.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/59/Goblet-smoothed-sharpened-top.png> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Wynnmc at en.wikibooks
- **File:Goblets_two_mooroon.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/58/Goblets_two_mooroon.png *License:* GFDL *Contributors:* Own work *Original artist:* Mooroon (Mooroon (talk) 16:13, 30 September 2012 (UTC))
- **File:Green(tutorialBlender).PNG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/41/Green%28tutorialBlender%29.PNG> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Vessesh Hebbar
- **File:Green_Jeep_With_Rocket_Launcher.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/18/Green_Jeep_With_Rocket_Launcher.png *License:* CC BY 3.0 *Contributors:* Own work *Original artist:* Telmer6
- **File:Green_ooze.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/56/Green_ooze.png *License:* CC BY-SA 1.0 *Contributors:* self *Original artist:* Blender Foundation
- **File:Green_ooze_props.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/da/Green_ooze_props.png *License:* CC BY-SA 1.0 *Contributors:* self *Original artist:* Blender Foundation
- **File:Greeneyes(1).jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/8d/Greeneyes%281%29.jpg> *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Chinmay gautam at English Wikibooks
- **File:Ground1.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/57/Ground1.jpg> *License:* CC BY 2.5 *Contributors:* ? *Original artist:* ?
- **File:HalfOutlineWineGlass.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/58/HalfOutlineWineGlass.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:HalfOutlineDrinkingGlass.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a1/HalfOutlineDrinkingGlass.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:HalfOutlineNapkinRing.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/8e/HalfOutlineNapkinRing.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:HaloFlareSunExample.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/85/HaloFlareSunExample.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:Handicon_blender_2.49a.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/78/Handicon_blender_2.49a.png *License:* Public domain *Contributors:* http://en.wikibooks.org/wiki/File:Handicon_blender_2.49a.png *Original artist:* b:en:User:Siggimund
- **File:Hoodstep1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/46/Hoodstep1.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Alexwill84 at English Wikibooks
- **File:Hoodstep2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a3/Hoodstep2.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Alexwill84 at English Wikibooks
- **File:Hoodstep3.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b8/Hoodstep3.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Alexwill84 at English Wikibooks
- **File:Hoodstep4.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5a/Hoodstep4.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Alexwill84 at English Wikibooks
- **File:Hoodstep5.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/43/Hoodstep5.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Alexwill84 at English Wikibooks

- **File:Hoodstep6.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/55/Hoodstep6.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Alexwill84 at English Wikibooks
- **File:Hoodstep7.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b6/Hoodstep7.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Alexwill84 at English Wikibooks
- **File:Hoodstep8.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/85/Hoodstep8.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Alexwill84 at English Wikibooks
- **File:House-complete.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/21/House-complete.png> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Wynnmcc at en.wikibooks
- **File:Ie_Arm_parent.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c8/Ie_Arm_parent.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Gabio at English Wikibooks
- **File:Ie_Ghost.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fa/Ie_Ghost.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_IK_1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/28/Ie_IK_1.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_IK_chain_Len.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8d/Ie_IK_chain_Len.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_IK_chain_Tip.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/31/Ie_IK_chain_Tip.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_IK_chain_limit.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8a/Ie_IK_chain_limit.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_IK_chain_rot.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/32/Ie_IK_chain_rot.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_IK_limit_buttons.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/60/Ie_IK_limit_buttons.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Gabio at English Wikibooks
- **File:Ie_Mult.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6f/Ie_Mult.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_Parent_bones.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/50/Ie_Parent_bones.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_Weight.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/50/Ie_Weight.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_arm_parent_menu.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/3e/Ie_arm_parent_menu.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_bboneyeinout.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/64/Ie_bboneyeinout.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_bbones.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f7/Ie_bbones.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_bonesname.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d8/Ie_bonesname.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_constraint.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/03/Ie_constraint.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Gabio at en.wikibooks
- **File:Ie_envelope.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d5/Ie_envelope.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_hinge.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/1e/Ie_hinge.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_little_zone.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/ab/Ie_little_zone.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_ok_zone.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/06/Ie_ok_zone.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_paint_tube.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/66/Ie_paint_tube.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_path.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/62/Ie_path.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_stick_guy.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/98/Ie_stick_guy.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks

- **File:Ie_track_cam.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/66/Ie_track_cam.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Ie_track_eye.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/56/Ie_track_eye.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Importing_Image.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6d/Importing_Image.jpg *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Badalia at English Wikibooks
- **File:In_Arms_Starting_Menu.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/dd/In_Arms_Starting_Menu.JPG *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Jeremy Buller
- **File:Information_icon4.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/1d/Information_icon4.svg *License:* Public domain *Contributors:* modified versions from below, which were modified from <http://www.kde-look.org/> *Original artist:* penubag (color adjustments)
- **File:Initial_weights.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/29/Initial_weights.jpg *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Stalepanda at English Wikibooks
- **File:InkscapePathResulttjb0607.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f7/InkscapePathResulttjb0607.png> *License:* CC BY 3.0 *Contributors:* Own work *Original artist:* Tjb0607
- **File:Inkscape_icons_edit_nodes.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e2/Inkscape_icons_edit_nodes.svg *License:* GPL *Contributors:* <http://www.inkscape.org/> *Original artist:* <http://www.inkscape.org/>
- **File:Inkscape_icons_node_add.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/bb/Inkscape_icons_node_add.svg *License:* GPL *Contributors:* Inkscape (original document) *Original artist:* Inkscape Developers
- **File:Inkscape_icons_tool_pointer.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e8/Inkscape_icons_tool_pointer.svg *License:* GPL *Contributors:* Inkscape (original document) *Original artist:* Inkscape Developers
- **File:Innertire1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/68/Innertire1.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* JonnyJack
- **File:Install_aAddon.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0c/Install_aAddon.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Iris.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5c/Iris.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was ThomasB at English Wikibooks
- **File:Isometric.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/38/Isometric.svg> *License:* Public domain *Contributors:* Transferred from en.wikibooks/ *Original artist:* Original uploader was BANZ111, at en.wikibooks
- **File:Jeep600detailmooroon.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1b/Jeep600detailmooroon.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Mooroon
- **File:Jeep600mooroon.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e6/Jeep600mooroon.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Mooroon
- **File:JeepRender.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/15/JeepRender.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* DDD3x
- **File:JeepSeatRendered.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c3/JeepSeatRendered.jpg> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Carpetsmoker at English Wikibooks
- **File:Jeepbody10.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/4a/Jeepbody10.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody11.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/6e/Jeepbody11.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody12.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/53/Jeepbody12.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody13.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d6/Jeepbody13.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody14.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1d/Jeepbody14.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody15.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1c/Jeepbody15.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody16.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/6c/Jeepbody16.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody17.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/24/Jeepbody17.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks

- **File:Jeepbody18.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/74/Jeepbody18.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody19.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/17/Jeepbody19.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody1_(2).jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fb/Jeepbody1_%282%29.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Jeepbody2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5c/Jeepbody2.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody20.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d8/Jeepbody20.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody21.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e8/Jeepbody21.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody22.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2d/Jeepbody22.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody4.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/52/Jeepbody4.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody5.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/02/Jeepbody5.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody6.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ac/Jeepbody6.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepbody9.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/ee/Jeepbody9.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepseat1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/68/Jeepseat1.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepseat10.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/ea/Jeepseat10.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Josellis at English Wikibooks
- **File:Jeepseat2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9e/Jeepseat2.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepseat3.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/4a/Jeepseat3.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepseat4.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/09/Jeepseat4.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepseat5.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a2/Jeepseat5.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepseat6.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/46/Jeepseat6.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepseat7.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/38/Jeepseat7.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jeepseat8.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/13/Jeepseat8.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:JewelExample.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/26/JewelExample.png> *License:* Public domain *Contributors:* own work (screenshot) *Original artist:* Michał Sapalski (username: sapalskimichal). The original uploader was Sapalskimichal at English Wikibooks
- **File:JewelFinished.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/90/JewelFinished.png> *License:* Public domain *Contributors:* own work (screenshot) *Original artist:* Michał Sapalski (username: sapalskimichal). The original uploader was Sapalskimichal at English Wikibooks

- **File:Jewel_Materials.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/4b/Jewel_Materials.png *License:* Public domain *Contributors:* own work (screenshot) *Original artist:* Michał Sapalski (username: sapalskimichal). The original uploader was Sapalskimichal at English Wikibooks
- **File:Johno_diag_1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/57/Johno_diag_1.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Johno at English Wikibooks
- **File:Johno_logocircle1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/16/Johno_logocircle1.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Johno at English Wikibooks
- **File:Johno_logocircle2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f6/Johno_logocircle2.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Johno at English Wikibooks
- **File:Johno_logocircle3.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a9/Johno_logocircle3.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Johno at English Wikibooks
- **File:Johno_logocircle4.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/be/Johno_logocircle4.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Johno at English Wikibooks
- **File:Johno_logofinished.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/55/Johno_logofinished.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Johno at English Wikibooks
- **File:Jrl1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d2/Jrl1.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jrl10.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c1/Jrl10.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jrl11.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/3e/Jrl11.png> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was AverieZen at en.wikibooks
- **File:Jrl8.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/dd/Jrl8.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Jrl9.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/ef/Jrl9.png> *License:* GPL *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was AverieZen at en.wikibooks
- **File:Komodo_Dragon_(2007).jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/55/Komodo_Dragon_%282007%29.jpg *License:* CC BY-SA 2.0 *Contributors:* <http://www.flickr.com/photos/33037982@N04/4181853927/> *Original artist:* Wallygrom
- **File:L01_default_quad.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f2/L01_default_quad.png *License:* CC BY-SA 3.0 *Contributors:* Screenshot of the GPL program Blender *Original artist:* MercedMike
- **File:L02_first_render.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c5/L02_first_render.png *License:* CC BY-SA 3.0 *Contributors:* Screenshot of the GPL program Blender showing point lighting *Original artist:* MercedMike
- **File:L09A_make_transparent.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/09/L09A_make_transparent.png *License:* CC BY-SA 3.0 *Contributors:* Screenshot of the GPL program Blender *Original artist:* MercedMike
- **File:L10_receive_transp.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/19/L10_receive_transp.png *License:* CC BY-SA 3.0 *Contributors:* Screenshot of the GPL program Blender *Original artist:* MercedMike
- **File:L11_rotations.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/97/L11_rotations.png *License:* CC BY-SA 3.0 *Contributors:* Screenshot of the GPL program Blender *Original artist:* MercedMike
- **File:LEMAtOrigin.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a8/LEMAtOrigin.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:LEMRotated45.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/00/LEMRotated45.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:LEMRotated45Translated.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2d/LEMRotated45Translated.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:LEMScaled50.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/ff/LEMScaled50.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:LEMSquashed50.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e9/LEMSquashed50.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:LEMTranslated.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c2/LEMTranslated.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:LEMTranslatedRotated45.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/be/LEMTranslatedRotated45.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:LSTV.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2a/LSTV.jpg> *License:* CC BY-SA 2.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Bootleg at English Wikibooks

- **File:Lattice_AddModifier.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9f/Lattice_AddModifier.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:Lattice_Animkeys.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/de/Lattice_Animkeys.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:Lattice_Context.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/4f/Lattice_Context.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:Lattice_Modifier.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f8/Lattice_Modifier.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:Lattice_Samples.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/48/Lattice_Samples.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:Lattice_UVWsets.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a9/Lattice_UVWsets.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:Lattice_UVWtypes.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a8/Lattice_UVWtypes.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:Lattice_complex.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/15/Lattice_complex.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:Lattice_eyes_with.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b8/Lattice_eyes_with.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:Lattice_eyes_without.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/2f/Lattice_eyes_without.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:Lattice_revkey.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/25/Lattice_revkey.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:Lattice_sliders.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/18/Lattice_sliders.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:Lava_seamless_texture.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9f/Lava_seamless_texture.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Layers(tutorialBlender).PNG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a0/Layers%28tutorialBlender%29.PNG> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Vessesh Hebbar
- **File:Layers_grid.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/3f/Layers_grid.png *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:GArantes *Original artist:* Lazy-lump at English Wikibooks
- **File:Lightprobe_hardware_diagram.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fe/Lightprobe_hardware_diagram.png *License:* CC-BY-SA-3.0 *Contributors:* No machine-readable source provided. Own work assumed (based on copyright claims). *Original artist:* No machine-readable author provided. Bmud assumed (based on copyright claims).
- **File:Limit-selection-to-visible.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/66/Limit-selection-to-visible.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Wynnmc at English Wikibooks
- **File:LinearTransformations.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/6d/LinearTransformations.svg> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:LinksPipeAddNew.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/cb/LinksPipeAddNew.png> *License:* GPL *Contributors:* Own work (Screenshot) *Original artist:* DDD3x
- **File:Logo_final.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/38/Logo_final.jpg *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Andrew Steinborn using CommonsHelper. *Original artist:* Kieran at English Wikibooks
- **File:Lone_House.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/39/Lone_House.jpg *License:* CC-BY-SA-3.0 *Contributors:* This file was created with Blender. *Original artist:* Michael Otto (user Mayqel)
- **File:LookForMapTo.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/69/LookForMapTo.png> *License:* GPL *Contributors:* Own work (Screenshot) *Original artist:* DDD3x
- **File:Main_menu_starting.JPG** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/7f/Main_menu_starting.JPG *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Jeremy B
- **File:MakeSeamlessSample.jpeg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/8c/MakeSeamlessSample.jpeg> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:MakeSeamlessSampleAfter.jpeg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/dc/MakeSeamlessSampleAfter.jpeg> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo

- **File:MakeSeamlessSampleUsage.jpeg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/02/MakeSeamlessSampleUsage.jpeg> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:MakeTrack.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ae/MakeTrack.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Totophe64 at English Wikibooks
- **File:Makehuman.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9b/Makehuman.jpg> *License:* CC0 *Contributors:* screenshot *Original artist:* othman ahmad
- **File:Makehuman_import.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/06/Makehuman_import.jpg *License:* CC BY-SA 3.0 *Contributors:* screenshot *Original artist:* Othman Ahmad
- **File:MapInputChange.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/bb/MapInputChange.png> *License:* GPL *Contributors:* Own work (Screenshot) *Original artist:* DDD3x
- **File:MapToChange.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/35/MapToChange.png> *License:* GPL *Contributors:* Own work (Screenshot) *Original artist:* DDD3x
- **File:Marking_The_Middle.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/43/Marking_The_Middle.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:MaterialChangePreview.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a2/MaterialChangePreview.png> *License:* GPL *Contributors:* Own work (Screenshot) *Original artist:* DDD3x
- **File:MaterialNodesRenderAfter.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/67/MaterialNodesRenderAfter.jpg> *License:* CC BY-SA 3.0 *Contributors:* Blender *Original artist:* Lawrence D' Oliveira
- **File:MaterialNodesRenderBefore.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/25/MaterialNodesRenderBefore.jpg> *License:* CC BY-SA 3.0 *Contributors:* Blender *Original artist:* Lawrence D' Oliveira
- **File:MaterialNodesSimpleGradation.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/19/MaterialNodesSimpleGradation.png> *License:* CC BY-SA 3.0 *Contributors:* Blender *Original artist:* Lawrence D' Oliveira
- **File:MaterialpicA..png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5d/MaterialpicA..png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:MaterialpicB.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c6/MaterialpicB.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:MaterialpicC.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ac/MaterialpicC.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:MaterialpicD..png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/4d/MaterialpicD..png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:Materials_button.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/50/Materials_button.png *License:* CC BY-SA 1.0 *Contributors:* self *Original artist:* Blender Foundation
- **File:MazeTutorial.01_grid-select.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/ba/MazeTutorial.01_grid-select.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Malaz at English Wikibooks
- **File:MazeTutorial.02_cone-extrude.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9c/MazeTutorial.02_cone-extrude.png *License:* Public domain *Contributors:* My own work. *Original artist:* Jackson Williams (Malaz). The original uploader was Malaz at English Wikibooks
- **File:MazeTutorial.03_cone-align.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d2/MazeTutorial.03_cone-align.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Malaz at English Wikibooks
- **File:MazeTutorial.04_character-logic.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/78/MazeTutorial.04_character-logic.png *License:* Public domain *Contributors:* This is a screenshot of my own work within Blender.; Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Jackson Williams (Malaz); Original uploader was Malaz at en.wikibooks
- **File:MazeTutorial.05_mazegame-test.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f9/MazeTutorial.05_mazegame-test.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Malaz at English Wikibooks
- **File:MazeTutorial.06_cube-touch.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/64/MazeTutorial.06_cube-touch.png *License:* Public domain *Contributors:* This is by own work, created with the open source application Blender.; Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Jackson Williams (Malaz); Original uploader was Malaz at en.wikibooks
- **File:MazeTutorial.08_brick-wall.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/1b/MazeTutorial.08_brick-wall.png *License:* Public domain *Contributors:* This is my own work. *Original artist:* Jackson Williams (Malaz). The original uploader was Malaz at English Wikibooks
- **File:MazeTutorial.09_no-textures.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/43/MazeTutorial.09_no-textures.png *License:* Public domain *Contributors:* My own work. *Original artist:* Jackson Williams (Malaz). The original uploader was Malaz at English Wikibooks
- **File:MazeTutorial.10_edit-material.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/22/MazeTutorial.10_edit-material.png *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Jackson Williams (Malaz). Original uploader was Malaz at en.wikibooks
- **File:MazeTutorial.11_with-materials.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/83/MazeTutorial.11_with-materials.png *License:* Public domain *Contributors:* My own work. *Original artist:* Jackson Williams (Malaz). The original uploader was Malaz at English Wikibooks

- **File:MazeTutorial.12_with-mist.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/4f/MazeTutorial.12_with-mist.png *License:* Public domain *Contributors:* My own work. *Original artist:* Jackson Williams (Malaz). The original uploader was Malaz at English Wikibooks
- **File:MazeTutorial.13_mist-buttons.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/66/MazeTutorial.13_mist-buttons.png *License:* Public domain *Contributors:* My own work. *Original artist:* Jackson Williams (Malaz)
- **File:MazeTutorial.14_maze-extruded.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/31/MazeTutorial.14_maze-extruded.png *License:* Public domain *Contributors:* My own work. *Original artist:* Jackson Williams (Malaz). The original uploader was Malaz at English Wikibooks
- **File:MazeTutorial.15_scene-button.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/44/MazeTutorial.15_scene-button.png *License:* Public domain *Contributors:* My own work. *Original artist:* Jackson Williams (Malaz)
- **File:Menu_icon_selector1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0e/Menu_icon_selector1.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Jeremy B
- **File:Mergefrom.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/0f/Mergefrom.svg> *License:* Public domain *Contributors:* ? *Original artist:* ?
- **File:MetallicMagentaBall.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/6f/MetallicMagentaBall.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:Modifier_editmode.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/56/Modifier_editmode.png *License:* GPL *Contributors:* Own work *Original artist:* Myself
- **File:Muescas_y_espacios.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b6/Muescas_y_espacios.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Reptiles extintos at English Wikibooks
- **File:NLA1.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1d/NLA1.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Zeviion at English Wikibooks
- **File:NLA2.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/35/NLA2.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Zeviion at English Wikibooks
- **File:NLA3.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a1/NLA3.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Zeviion at English Wikibooks
- **File:NLA4.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/94/NLA4.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Zeviion at English Wikibooks
- **File:NLA5.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a6/NLA5.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Zeviion at English Wikibooks
- **File:NLA6.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/85/NLA6.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Zeviion at English Wikibooks
- **File:Name_drawing.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/58/Name_drawing.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Gabio at en.wikibooks
- **File:Next_new_button.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/16/Next_new_button.png *License:* GPL *Contributors:* self *Original artist:* Blender Foundation
- **File:Nibelungendrache.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/34/Nibelungendrache.jpg> *License:* Public domain *Contributors:* ? *Original artist:* ?
- **File:Nla-2.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/6b/Nla-2.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Halley at English Wikibooks
- **File:Nla-3.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/bf/Nla-3.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Halley at English Wikibooks
- **File:Nla-4.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/6f/Nla-4.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Halley at English Wikibooks
- **File:Node_Tree_Types.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c6/Node_Tree_Types.png *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Win Bigly
- **File:Nofaces.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d1/Nofaces.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* JonnyJack
- **File:Noisevertices.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/32/Noisevertices.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* SergeantOreo at English Wikibooks
- **File:Nuvola_apps_bookcase.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a5/Nuvola_apps_bookcase.svg *License:* LGPL *Contributors:* The source code of this SVG is <a data-x-rel='nofollow' class='external text' href='//validator.w3.org/check?uri=https%3A%2F%2Commons.wikimedia.org%2Fwiki%2FSpecial%3AFilepath%2FNuvola_apps_bookcase.svgss=1#source>valid. *Original artist:* Peter Kemp
- **File:Nuvola_apps_important_yellow.svg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/dc/Nuvola_apps_important_yellow.svg *License:* LGPL *Contributors:* An icon from gnome-themes-extras-0.9.0.tar.bz2 (specifically Nuvola/icons/scalable/emblems/emblem-important.svg) by David Vignoni. *Original artist:* Modified to look more like the PNG file by Bastique. Recolored by flamura.
- **File:Nuvola_apps_kwwrite.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/71/Nuvola_apps_kwwrite.png *License:* LGPL *Contributors:* http://icon-king.com *Original artist:* David Vignoni / ICON KING
- **File:OOPS_bouncing.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0c/OOPS_bouncing.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Gabio at English Wikibooks

- **File:ObjectEdit_select_1.gif** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/7e/ObjectEdit_select_1.gif *License:* GPL
Contributors: Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Robpers2003 at English Wikibooks
- **File:Object_settings(tutorialBlender).PNG** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fc/Object_settings%28tutorialBlender%29.PNG *License:* CC0 *Contributors:* Own work *Original artist:* Vesseshhebar
- **File:Objectmodeverification.PNG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/fb/Objectmodeverification.PNG> *License:* GPL *Contributors:* Screenshot taken by Jake.lewis4 *Original artist:* Jake.lewis4
- **File:Objects_bent_initial.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/73/Objects_bent_initial.jpg *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Stakepanda at English Wikibooks
- **File:Obstacle-basic-fluid-tutorial-blender2.76.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/7f/Obstacle-basic-fluid-tutorial-blender2.76.jpg> *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Octahedron_bone.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/69/Octahedron_bone.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Gabio at en.wikibooks
- **File:Oneleg-ginger.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/15/Oneleg-ginger.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Darkone at en.wikibooks. Later version(s) were uploaded by Argento at en.wikibooks.
- **File:OriginalOutlineInBlender.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e5/OriginalOutlineInBlender.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:Orthographic.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/11/Orthographic.svg> *License:* Public domain *Contributors:* <http://en.wikibooks.org/wiki/File:Orthographic.svg> *Original artist:* b:en:User:BANZ111
- **File:OrthographicProjection.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/65/OrthographicProjection.svg> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:OutlineSpunInBlender1.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ac/OutlineSpunInBlender1.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:OutlineSpunInBlender2.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/83/OutlineSpunInBlender2.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:OutlineSpunInBlender3.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/db/OutlineSpunInBlender3.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender 3d Screen Shots
- **File:OutlineSpunInBlender4.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/55/OutlineSpunInBlender4.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:Outsideedge.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/15/Outsideedge.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* JonnyJack
- **File:Paint_panel.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/1c/Paint_panel.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Gabio at English Wikibooks
- **File:Peng.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/15/Peng.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Jesse S. (Bnty)
- **File:Penguin.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/88/Penguin.jpg> *License:* Public domain *Contributors:* No machine-readable source provided. Own work assumed (based on copyright claims). *Original artist:* No machine-readable author provided. WhiplashEffigy assumed (based on copyright claims).
- **File:Penguin_from_sphere_-_wing_start.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e8/Penguin_from_sphere_-_wing_start.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Romain Behar
- **File:Penguin_orbisonitrum.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/ac/Penguin_orbisonitrum.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Orbisonitrum at English Wikibooks
- **File:Perspective-1point.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/ef/Perspective-1point.svg> *License:* Public domain *Contributors:* Own work (Marco Polo)
Original artist: Marco Polo at en.wikipedia
- **File:Perspective-2point.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/7e/Perspective-2point.svg> *License:* Public domain *Contributors:* Own work (Marco Polo) *Original artist:* Marco Polo at English Wikipedia
- **File:Perspective-3point.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/aa/Perspective-3point.svg> *License:* Public domain *Contributors:* Own work (Marco Polo)
Original artist: Marco Polo at en.wikipedia
- **File:PerspectiveProjection.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/12/PerspectiveProjection.svg> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:Pixareyes-Eye_mesh.gif** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/14/Pixareyes-Eye_mesh.gif *License:* Public domain *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was WeirdHat at en.wikibooks. Later version(s) were uploaded by Rdp at en.wikibooks.
- **File:Pixareyes-iris_blueeye.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/da/Pixareyes-iris_blueeye.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Hrakaroo at English Wikibooks
- **File:Pixareyes-iris_greeneye.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/1d/Pixareyes-iris_greeneye.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Hrakaroo at English Wikibooks

- **File:Pixareyes-iris_redeye.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/87/Pixareyes-iris_redeye.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Hrakaroo at English Wikibooks
- **File:Pixareyes-iriscoloredit.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f8/Pixareyes-iriscoloredit.png> *License:* GPL *Contributors:* Transferred from en.wikibooks *Original artist:* Original uploader was Hrakaroo at en.wikibooks
- **File:PlastickyMagentaBall.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9f/PlastickyMagentaBall.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:PolyByPoly_Continue.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ef/PolyByPoly_Continue.jpg *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:PolyByPoly_Finish.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/cf/PolyByPoly_Finish.jpg *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:Poly_loopPlanning.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/54/Poly_loopPlanning.jpg *License:* CC-BY-SA-3.0 *Contributors:*
- TNMSittingBuddhaFace.jpg *Original artist:*
- derivative work: Rosver (talk)
- **File:Pose_panel.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/37/Pose_panel.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adriogola using CommonsHelper. *Original artist:* The original uploader was Gabio at English Wikibooks
- **File:Procedural_eyeball_blender2.75_1-2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/00/Procedural_eyeball_blender2.75_1-2.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/ba/Procedural_eyeball_blender2.75_1.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_10.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/4a/Procedural_eyeball_blender2.75_10.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_11.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ec/Procedural_eyeball_blender2.75_11.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_13-1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/4e/Procedural_eyeball_blender2.75_13-1.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_13-2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f7/Procedural_eyeball_blender2.75_13-2.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_14-1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d8/Procedural_eyeball_blender2.75_14-1.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_14-2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5d/Procedural_eyeball_blender2.75_14-2.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_15.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5e/Procedural_eyeball_blender2.75_15.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_16,5.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/09/Procedural_eyeball_blender2.75_16%2C5.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_16-A2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6d/Procedural_eyeball_blender2.75_16-A2.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_16-B.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/19/Procedural_eyeball_blender2.75_16-B.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_17.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/ec/Procedural_eyeball_blender2.75_17.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_18.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/55/Procedural_eyeball_blender2.75_18.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_19.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/cd/Procedural_eyeball_blender2.75_19.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_2-2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8e/Procedural_eyeball_blender2.75_2-2.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a0/Procedural_eyeball_blender2.75_2.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_20-2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/92/Procedural_eyeball_blender2.75_20-2.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_21-1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/2a/Procedural_eyeball_blender2.75_21-1.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_21-2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6e/Procedural_eyeball_blender2.75_21-2.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_22-1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/71/Procedural_eyeball_blender2.75_22-1.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Procedural_eyeball_blender2.75_22-2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/8/8c/Procedural_eyeball_blender2.75_22-2.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser

- **File:Procedural_eyeball_blender2.75_23.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/8/84/Procedural_eyeball_blender2.75_23.jpg License: CC BY-SA 4.0 Contributors: Own work Original artist: Animajosser
- **File:Procedural_eyeball_blender2.75_24.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/5/5f/Procedural_eyeball_blender2.75_24.jpg License: CC BY-SA 4.0 Contributors: Own work Original artist: Animajosser
- **File:Procedural_eyeball_blender2.75_25.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/3/3f/Procedural_eyeball_blender2.75_25.jpg License: CC BY-SA 4.0 Contributors: Own work Original artist: Animajosser
- **File:Procedural_eyeball_blender2.75_26.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/7/7a/Procedural_eyeball_blender2.75_26.jpg License: CC BY-SA 4.0 Contributors: Own work Original artist: Animajosser
- **File:Procedural_eyeball_blender2.75_3.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/0/0b/Procedural_eyeball_blender2.75_3.jpg License: CC BY-SA 4.0 Contributors: Own work Original artist: Animajosser
- **File:Procedural_eyeball_blender2.75_4.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/2/26/Procedural_eyeball_blender2.75_4.jpg License: CC BY-SA 4.0 Contributors: Own work Original artist: Animajosser
- **File:Procedural_eyeball_blender2.75_5.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/9/9b/Procedural_eyeball_blender2.75_5.jpg License: CC BY-SA 4.0 Contributors: Own work Original artist: Animajosser
- **File:Procedural_eyeball_blender2.75_6.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/f/fc/Procedural_eyeball_blender2.75_6.jpg License: CC BY-SA 4.0 Contributors: Own work Original artist: Animajosser
- **File:Procedural_eyeball_blender2.75_7.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/9/98/Procedural_eyeball_blender2.75_7.jpg License: CC BY-SA 4.0 Contributors: Own work Original artist: Animajosser
- **File:Procedural_eyeball_blender2.75_8.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/4/4d/Procedural_eyeball_blender2.75_8.jpg License: CC BY-SA 4.0 Contributors: Own work Original artist: Animajosser
- **File:Procedural_eyeball_blender2.75_9.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/a/a7/Procedural_eyeball_blender2.75_9.jpg License: CC BY-SA 4.0 Contributors: Own work Original artist: Animajosser
- **File:Procedural_eyeball_blender2.75_if_rendered.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/b/ba/Procedural_eyeball_blender2.75_if_rendered.jpg License: CC BY-SA 4.0 Contributors: Own work Original artist: Animajosser
- **File:Procedural_eyeball_blender2.75_rendered.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/1/1a/Procedural_eyeball_blender2.75_rendered.jpg License: CC BY-SA 4.0 Contributors: Own work Original artist: Animajosser
- **File:Psychedelic_Jeep.png** Source: https://upload.wikimedia.org/wikipedia/commons/7/72/Psychedelic_Jeep.png License: CC BY-SA 3.0 Contributors: Own work Original artist: MercedMike
- **File:RUSTY_BALL.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/a/a1/RUSTY_BALL.jpg License: CC BY-SA 3.0 Contributors: Own work Original artist: MercedMike
- **File:RainbowTextureNodesRender.png** Source: <https://upload.wikimedia.org/wikipedia/commons/6/6b/RainbowTextureNodesRender.png> License: CC BY-SA 3.0 Contributors: Blender Screenshot Original artist: Lawrence D'Oliveiro
- **File:Rechte-hand-regel.jpg** Source: <https://upload.wikimedia.org/wikipedia/commons/7/79/Rechte-hand-regel.jpg> License: CC-BY-SA-3.0 Contributors: Own work Original artist: Abdull
- **File:Red_Wine_Glas.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/c/c0/Red_Wine_Glass.jpg License: CC-BY-SA-3.0 Contributors: Own work Original artist: André Karwath aka Aka
- **File:Redeyes.jpg** Source: <https://upload.wikimedia.org/wikipedia/commons/c/cd/Redeyes.jpg> License: CC BY-SA 3.0 Contributors: Transferred from en.wikibooks to Commons. Original artist: Chinmay gautam at English Wikibooks
- **File:RenderLayer1-Settings_BlenderMistTutorial.jpg** Source: https://upload.wikimedia.org/wikipedia/commons/e/e9/RenderLayer1-Settings_BlenderMistTutorial.jpg License: GPL Contributors: Blender Original artist: ?
- **File:RenderResult.jpg** Source: <https://upload.wikimedia.org/wikipedia/commons/0/0d/RenderResult.jpg> License: CC BY-SA 3.0 Contributors: Transferred from en.wikibooks to Commons. Original artist: Radialronnie at English Wikibooks
- **File:RenderScene-20081103151408.png** Source: <https://upload.wikimedia.org/wikipedia/commons/5/5d/RenderScene-20081103151408.png> License: CC BY-SA 3.0 Contributors: Transferred from en.wikibooks to Commons. Original artist: Radialronnie at English Wikibooks
- **File:Render_Backlight.png** Source: https://upload.wikimedia.org/wikipedia/commons/5/56/Render_Backlight.png License: Public domain Contributors: Transferred from en.wikibooks to Commons. Original artist: Radialronnie at English Wikibooks
- **File:Render_Layers_tab.png** Source: https://upload.wikimedia.org/wikipedia/commons/c/cb/Render_Layers_tab.png License: GPL Contributors: Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. Original artist: The original uploader was Soeb at English Wikibooks
- **File:Render_SphericalBlend.png** Source: https://upload.wikimedia.org/wikipedia/commons/6/61/Render_SphericalBlend.png License: Public domain Contributors: Transferred from en.wikibooks to Commons. Original artist: Radialronnie at English Wikibooks
- **File:Render_tab.png** Source: https://upload.wikimedia.org/wikipedia/commons/d/d4/Render_tab.png License: GPL Contributors: Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. Original artist: The original uploader was Soeb at English Wikibooks
- **File:Rendered_View.png** Source: https://upload.wikimedia.org/wikipedia/commons/1/13/Rendered_View.png License: GPL Contributors: Blender 2.78a Original artist: Blender 2.78a
- **File:Rendered_spun_goblet.png** Source: https://upload.wikimedia.org/wikipedia/commons/5/56/Rendered_spun_goblet.png License: CC BY-SA 3.0 Contributors: Own work Original artist: Arch dude
- **File:Rendertire.png** Source: <https://upload.wikimedia.org/wikipedia/commons/1/18/Rendertire.png> License: CC BY-SA 3.0 Contributors: Own work Original artist: JonnyJack

- **File:Rig_Step1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a5/Rig_Step1.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Wavez at English Wikibooks
- **File:RingBar1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/8b/RingBar1.png> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was Darkonc at English Wikibooks
- **File:RingBar2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f9/RingBar2.png> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was Darkonc at English Wikibooks
- **File:RingBar3.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/02/RingBar3.png> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was Darkonc at English Wikibooks
- **File:RingBar4.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/74/RingBar4.png> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was Darkonc at English Wikibooks
- **File:Ripple_disc_norm.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c4/Ripple_disc_norm.jpg *License:* Public domain *Contributors:* Blender 3D Render *Original artist:* Jeremy B.. The original uploader was Bullercruz1 at English Wikibooks
- **File:Rollingball.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/cd/Rollingball.jpg> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Othman Ahmad
- **File:Rotate1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/4b/Rotate1.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* JonnyJack
- **File:Rotatetwo.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/86/RotateTwo.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* JonnyJack
- **File:RotationAxisDirections.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a3/RotationAxisDirections.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:Rust_and_dirt.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/bb/Rust_and_dirt.jpg *License:* CC-BY-SA-3.0 *Contributors:* Picture taken and uploaded by Roger McLassus. *Original artist:* Roger McLassus
- **File:Scaledgrid.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d8/Scaledgrid.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* SergeantOreo at English Wikibooks
- **File:Screen_Shot_11-29-16_at_09.47_PM.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/ba/Screen_Shot_11-29-16_at_09.47_PM.png *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* LightspeedLife
- **File:Screen_Shot_11-29-16_at_11.12_PM.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/1f/Screen_Shot_11-29-16_at_11.12_PM.png *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* LightspeedLife
- **File:Sea-water-setting.gif** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/3c/Sea-water-setting.gif> *License:* GPL *Contributors:* Screenshot of the GPL program Blender *Original artist:* Weeix (talk) 14:15, 2 November 2010 (UTC)
- **File:Seamless_grass.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5f/Seamless_grass.jpg *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* MercedMike
- **File:Second-cut-removed.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c2/Second-cut-removed.png> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Wynnmc at English Wikibooks
- **File:Select_image.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/3e>Select_image.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Goeland86 at English Wikibooks
- **File:Selectfaces.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c5/Selectfaces.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* JonnyJack
- **File:Selecting-vertices.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/3e>Selecting-vertices.png> *License:* Public domain *Contributors:* Desktop screen capture, my PC at home *Original artist:* Paul James Chatman
- **File:Selection_Scale_Vertices.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/79/Selection_Scale_Vertices.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Selection_Upper_Vertices.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fe/Selection_Upper_Vertices.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:SetAAO-2008110322151223.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/37/SetAAO-2008110322151223.png> *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Radialronnie at English Wikibooks
- **File:SetWireDrawType.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/36/SetWireDrawType.png> *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Radialronnie at English Wikibooks
- **File:Settings.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/91/Settings.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Totophe64 at English Wikibooks
- **File:Settings1.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e2/Settings1.jpg> *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Chinmay gautam at English Wikibooks
- **File:Shadersettings.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/49/Shadersettings.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* SergeantOreo at English Wikibooks
- **File:ShapeInArbitraryView.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a9/ShapeInArbitraryView.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:ShapeInTopView.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c7/ShapeInTopView.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation

- **File:Shape_Key_Editor.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/2e/Shape_Key_Editor.png *License:* GPL *Contributors:* Screenshot of the GPL program Blender 2.78 *Original artist:* Chloris Pale Green
- **File:Shape_panel.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/1b/Shape_panel.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Gabio at English Wikibooks
- **File>Showpointer.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e1/Showpointer.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Jeremy B
- **File:Sides_of_the_tire.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/7/74/Sides_of_the_tire.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Silver_goblet.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a7/Silver_goblet.jpg *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks; transferred to Commons by User:Adrignola using CommonsHelper. *Original artist:* Original uploader was Clone Dad at en.wikibooks
- **File:SimpleFreestylePetals.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/7b/SimpleFreestylePetals.png> *License:* CC0 *Contributors:* Own work *Original artist:* Ldo
- **File:SimpleMan1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d8/SimpleMan1.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan10.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ab/SimpleMan10.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan11.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/30/SimpleMan11.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan12.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1d/SimpleMan12.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan13.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ae/SimpleMan13.png> *License:* CC0 *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan14.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a6/SimpleMan14.png> *License:* CC0 *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan15.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/55/SimpleMan15.png> *License:* CC0 *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan16.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/92/SimpleMan16.png> *License:* CC0 *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f7/SimpleMan2.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan3.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a7/SimpleMan3.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan4.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5c/SimpleMan4.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan5.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2a/SimpleMan5.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan6.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/89/SimpleMan6.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan7.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/96/SimpleMan7.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan8.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/73/SimpleMan8.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:SimpleMan9.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/63/SimpleMan9.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:SimpwheelpicA.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1b/SimpwheelpicA.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:SimpwheelpicB.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/0b/SimpwheelpicB.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:SimpwheelpicC.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/41/SimpwheelpicC.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:Simulation-basic-fluid-tutorial-blender2.76.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/18/Simulation-basic-fluid-tutorial-blender2.76.jpg> *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:SizeTurbulence.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2f/SizeTurbulence.png> *License:* GPL *Contributors:* Own work (Screenshot) *Original artist:* DDD3x
- **File:Skateboard-example.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9a/Skateboard-example.jpg> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Halley at English Wikibooks
- **File:Skybox_ground.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/ac/Skybox_ground.png *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Greensweater at English Wikibooks
- **File:Skybox_sky.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/06/Skybox_sky.png *License:* Public domain *Contributors:* (Original text): Used from <http://www.philohome.com> (free use for any purpose as stated on web page) *Original artist:* The original uploader was Greensweater at English Wikibooks

- **File:Soft+wind5.JPG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/8e/Soft%2Bwind5.JPG> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Eb264 at English Wikibooks
- **File:Soft+wind6.JPG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/97/Soft%2Bwind6.JPG> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Eb264 at English Wikibooks
- **File:Soft+wind7.JPG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f1/Soft%2Bwind7.JPG> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Eb264 at English Wikibooks
- **File:Soft+wind8.JPG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2f/Soft%2Bwind8.JPG> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Eb264 at English Wikibooks
- **File:Soft+wind9.JPG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/40/Soft%2Bwind9.JPG> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Eb264 at English Wikibooks
- **File:Soft_body_wind.blender_2.76.gif** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5a/Soft_body_wind.blender_2.76.gif *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Sphere_UVs_edited.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5e/Sphere_UVs_edited.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Sphere_projection.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/5c/Sphere_projection.png *License:* GPL *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Sphere_spherical_mapped_render.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/2/28/Sphere_spherical_mapped_render.png *License:* CC0 *Contributors:* Self-taken *Original artist:* Blender Foundation
- **File:Sphere_with_marked_seams.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/be/Sphere_with_marked_seams.png *License:* CC0 *Contributors:* Self-taken. *Original artist:* Blender Foundation
- **File:Spin_a_goblet.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f6/Spin_a_goblet.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Arch dude
- **File:Squaretriangle.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/99/Squaretriangle.png> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Lirferk at English Wikibooks
- **File:Squish_the_middle.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d1/Squish_the_middle.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Start_Element.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/0c/Start_Element.jpg *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:Start_ImportantDetails.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/14/Start_ImportantDetails.png *License:* GPL *Contributors:* Own work *Original artist:* Blender Foundation
- **File:Start_importantLoops.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d7/Start_importantLoops.png *License:* Public domain *Contributors:* Own work *Original artist:* blender foundation
- **File:Step1-base-shader.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/10/Step1-base-shader.jpg> *License:* CC BY-SA 2.5 *Contributors:* Own work *Original artist:* en:Dan13
- **File:Step1-scr.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c6/Step1-scr.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dan13 at English Wikibooks
- **File:StepFive.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/83/StepFive.png> *License:* Public domain *Contributors:* own work (screenshot) *Original artist:* Michał Sapalski (username: sapalskimichal). The original uploader was Sapalskimichal at English Wikibooks
- **File:StepFour.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1a/StepFour.png> *License:* Public domain *Contributors:* own work (screenshot) *Original artist:* Michał Sapalski (username: sapalskimichal). The original uploader was Sapalskimichal at English Wikibooks
- **File:StepOne.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9f/StepOne.png> *License:* Public domain *Contributors:* own work (screenshot) *Original artist:* Michał Sapalski (username: sapalskimichal). The original uploader was Sapalskimichal at English Wikibooks
- **File:StepThree.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/be/StepThree.png> *License:* Public domain *Contributors:* own work (screenshot) *Original artist:* Michał Sapalski (username: sapalskimichal). The original uploader was Sapalskimichal at English Wikibooks
- **File:StepTwo.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9a/StepTwo.png> *License:* Public domain *Contributors:* own work (screenshot) *Original artist:* Michał Sapalski (username: sapalskimichal). The original uploader was Sapalskimichal at English Wikibooks
- **File:Stick_bone.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c5/Stick_bone.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Gabio at English Wikibooks
- **File:Stretch-to_constraint.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/93/Stretch-to_constraint.jpg *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* SergeantOreo at English Wikibooks
- **File:StucciType.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9e/StucciType.png> *License:* GPL *Contributors:* Own work (Screenshot) *Original artist:* DDD3x
- **File:SubDivide_Smooth.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c4/SubDivide_Smooth.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:Subdivide_options_Blender_2.72.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a9/Subdivide_options_Blender_2.72.png *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Rxwxrxwrx
- **File:Subsurf-bad.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/ec/Subsurf-bad.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Arch dude

- **File:Subsurf-good.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/89/Subsurf-good.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Arch dude
- **File:Suzanne_as_anaglyph_rendering_on_Blender.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d4/Suzanne_as_anaglyph_rendering_on_Blender.jpg *License:* GFDL *Contributors:* Own work *Original artist:* Fabrice TIERCELIN
- **File:Sv1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5c/Sv1.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Sv2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/3a/Sv2.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Sv3.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/80/Sv3.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Sv4.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9d/Sv4.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Sv5.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a0/Sv5.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Sv6.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/90/Sv6.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:SwanChairSceneInBlender.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/75/SwanChairSceneInBlender.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:SwanChairSceneRenderedWithBlender.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/dc/SwanChairSceneRenderedWithBlender.jpg> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:SyncJaw.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/70/SyncJaw.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* AndyD at English Wikibooks
- **File:SyncLoops.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/bb/SyncLoops.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* AndyD at English Wikibooks
- **File:SyncSample.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/3e/SyncSample.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* AndyD at English Wikibooks
- **File:SyncSeqEditor.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f4/SyncSeqEditor.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:SyncSliderKeys.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/de/SyncSliderKeys.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:SyncSliders.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e6/SyncSliders.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:SyncSoundblock.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b2/SyncSoundblock.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:SyncSubSurf.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/69/SyncSubSurf.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:SyncTimeline.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/59/SyncTimeline.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* The original uploader was AndyD at English Wikibooks
- **File:TNMStandingBuddhaFace.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/45/TNMStandingBuddhaFace.jpg> *License:* CC-BY-SA-3.0 *Contributors:* No machine-readable source provided. Own work assumed (based on copyright claims). *Original artist:* No machine-readable author provided. World Imaging assumed (based on copyright claims).
- **File:TPieceLoops.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/73/TPieceLoops.png> *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:Texture1-mat-panel.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/92/Texture1-mat-panel.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dan13 at English Wikibooks
- **File:Texture1-tex-panel.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/51/Texture1-tex-panel.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dan13 at English Wikibooks
- **File:Texture2-mat-panel.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/a6/Texture2-mat-panel.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dan13 at English Wikibooks
- **File:Texture2-render.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/39/Texture2-render.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dan13 at English Wikibooks
- **File:Texture2-tex-panel.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/64/Texture2-tex-panel.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dan13 at English Wikibooks
- **File:Texture3-mat-panel.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e5/Texture3-mat-panel.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dan13 at English Wikibooks

- **File:Texture3-render.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/dc/Texture3-render.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dan13 at English Wikibooks
- **File:Texture3-tex-panel.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/90/Texture3-tex-panel.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dan13 at English Wikibooks
- **File:Texture4-mat-panel.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/ee/Texture4-mat-panel.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dan13 at English Wikibooks
- **File:Texture4-tex-panel.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/82/Texture4-tex-panel.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dan13 at English Wikibooks
- **File:TextureAddNew.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ac/TextureAddNew.png> *License:* GPL *Contributors:* Own work (Screenshot) *Original artist:* DDDD3x
- **File:Texture_Bake_MapTo.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b7/Texture_Bake_MapTo.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:Texture_Bake_Material.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/01/Texture_Bake_Material.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:Texture_Bake_Preview.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/64/Texture_Bake_Preview.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:Texture_material_tabs.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6a/Texture_material_tabs.jpg *License:* Public domain *Contributors:* Blender 3D snapshot *Original artist:* Jeremy B.. The original uploader was Bullercruz1 at English Wikibooks
- **File:Texture_select2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b7/Texture_select2.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:The-rusty-ball.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/03/The-rusty-ball.png> *License:* CC0 *Contributors:* Own work *Original artist:* Weeix
- **File:Tiretut10.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/20/Tiretut10.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Tiretut11.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f6/Tiretut11.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Tiretut12.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/8d/Tiretut12.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Tiretut3.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/19/Tiretut3.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Tiretut4.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/41/Tiretut4.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Tiretut5.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5e/Tiretut5.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Tiretut6.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/7d/Tiretut6.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Tiretut7.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2f/Tiretut7.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Tiretut9.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/77/Tiretut9.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Toetsenbord_numeriek.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/37/Toetsenbord_numeriek.jpg *License:* CC BY-SA 2.5 *Contributors:* ? *Original artist:* ?
- **File:Toggle-nla-pose.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/7d/Toggle-nla-pose.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Halley at English Wikibooks
- **File:Toggle-nla-strips.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/df/Toggle-nla-strips.jpg> *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Halley at English Wikibooks
- **File:Tools_shading.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/d/d2/Tools_shading.png *License:* GPL *Contributors:* Blender v2.78 *Original artist:* Bobhaspants (talk) 18:20, 17 March 2017 (UTC)
- **File:Toywolf-construction-front.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5d/Toywolf-construction-front.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dmwick at English Wikibooks

- **File:Toywolf-construction-left.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1c/Toywolf-construction-left.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dmwick at English Wikibooks
- **File:Toywolf-construction-top.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/7/75/Toywolf-construction-top.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dmwick at English Wikibooks
- **File:Toywolf-front.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/3f/Toywolf-front.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dmwick at English Wikibooks
- **File:Toywolf-left.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/44/Toywolf-left.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dmwick at English Wikibooks
- **File:Toywolf-top.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/f9/Toywolf-top.jpg> *License:* CC BY-SA 2.5 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Dmwick at English Wikibooks
- **File:Treadtype1picA.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/cc/Treadtype1picA.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:Treadtype1picB.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/ca/Treadtype1picB.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:Treadtype2picA.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/6c/Treadtype2picA.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:Treadtype2picB.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/94/Treadtype2picB.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:Treadtype2picC.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2b/Treadtype2picC.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:Treadtype3picA.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/54/Treadtype3picA.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:Treadtype3picC.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9a/Treadtype3picC.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:Treadtype3picD.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/03/Treadtype3picD.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:TriangleArrow-Left.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/16/TriangleArrow-Left.svg> *License:* Public domain *Contributors:* ? *Original artist:* ?
- **File:TriangleArrow-Right.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/24/TriangleArrow-Right.svg> *License:* Public domain *Contributors:* ? *Original artist:* ?
- **File:Trianglexemplar.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/8b/Trianglexemplar.png> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Lirtferk at English Wikibooks
- **File:Tut.gif** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d5/Tut.gif> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Aidan.Sullivan at English Wikibooks
- **File:Tut.sb.1.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5f/Tut.sb.1.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Porlando at English Wikibooks
- **File:Tut.sb.2.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b5/Tut.sb.2.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Porlando at English Wikibooks
- **File:Tut.sb.3.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/4/44/Tut.sb.3.jpg> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Porlando at English Wikibooks
- **File:Tut.sb.4.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9d/Tut.sb.4.jpg> *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Porlando at English Wikibooks
- **File:TutFlipNorms-20081103140255.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/53/TutFlipNorms-20081103140255.png> *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Radialronnie at English Wikibooks
- **File:Two_objects.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/64/Two_objects.jpg *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Stalepanda at English Wikibooks
- **File:Twoleg-ginger-head.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9d/Twoleg-ginger-head.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Darkonc at English Wikibooks
- **File:Twoleg-ginger-pulled.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/5/5e/Twoleg-ginger-pulled.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Darkonc at English Wikibooks
- **File:Twoleg-ginger.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/8/89/Twoleg-ginger.png> *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Darkonc at English Wikibooks
- **File:UFO_Moonbase_Exterior_+Fireflies.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/32/UFO_Moonbase_Exterior_%2B_Fireflies.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo

- **File:UFO_Moonbase_Exterior_-_Fireflies.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c5/UFO_Moonbase_Exterior_-_Fireflies.png *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Ldo
- **File:UV_Finished.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/a/a0/UV_Finished.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:UV_Layout_Edit_A.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/1b/UV_Layout_Edit_A.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:UV_Layout_Edit_B.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b5/UV_Layout_Edit_B.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:UV_Seams.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/4b/UV_Seams.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:UV_SubSurf.gif** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/61/UV_SubSurf.gif *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:UV_Untangling.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b3/UV_Untangling.jpg *License:* Public domain *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:UV_Unwrap.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/3d/UV_Unwrap.jpg *License:* CC BY-SA 3.0 *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Quantum Anomaly at English Wikibooks
- **File:UV_sphere.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/5/55/UV_sphere.png *License:* CC0 *Contributors:* Own work *Original artist:* JamesNZ
- **File:UV_unwrapped_sphere.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/e/e9/UV_unwrapped_sphere.png *License:* CC0 *Contributors:* Self-taken. *Original artist:* Blender Foundation
- **File:UVsphere(tutorialBlender).PNG** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/61/UVsphere%28tutorialBlender%29.PNG> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Vessesh Hebbar
- **File:Uv_tool_Daniel_Banasik.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/6a/Uv_tool_Daniel_Banasik.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Vertex_groups.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/61/Vertex_groups.jpg *License:* CC-BY-SA-3.0 *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* Stalepanda at English Wikibooks
- **File:Vgroups_but.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f9/Vgroups_but.jpg *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was Gabio at English Wikibooks
- **File:Wallpaper_Particle_Systems,_Blender.gif** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/bd/Wallpaper_Particle_Systems%2C_Blender.gif *License:* CC0 *Contributors:* Own work *Original artist:* Vessesh Hebbar
- **File:Wheel1.jpg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/d9/Wheel1.jpg> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:Wheelrename.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/e/e4/Wheelrename.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:Wheelsubdivide.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/c/c1/Wheelsubdivide.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:WheelsubdivideB.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/2/2d/WheelsubdivideB.png> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* Buttflyhoney
- **File:Wikibooks-logo.svg** *Source:* <https://upload.wikimedia.org/wikipedia/commons/f/fa/Wikibooks-logo.svg> *License:* CC BY-SA 3.0 *Contributors:* Own work *Original artist:* User:Bastique, User:Ramac et al.
- **File:Wikipedia-logo.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/6/63/Wikipedia-logo.png> *License:* GFDL *Contributors:* based on the first version of the Wikipedia logo, by Nohat. *Original artist:* version 1 by Nohat (concept by Paullusmagnus);
- **File:WineglasTut_outline.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/30/WineglasTut_outline.png *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:WineglasTut_spin_frontview.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f4/WineglasTut_spin_frontview.jpg *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:WineglassTut_preparation.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/4/47/WineglassTut_preparation.jpg *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:WineglassTut_spin_settings.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/6/63/WineglassTut_spin_settings.jpg *License:* Public domain *Contributors:* Own work *Original artist:* Blender Foundation
- **File:Winglass_noTex.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f0/Winglass_noTex.jpg *License:* Public domain *Contributors:* Own work *Original artist:* Rosver
- **File:Wolf_building_front.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/1/18/Wolf_building_front.jpg *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Norleaf at English Wikibooks
- **File:Wolf_building_side.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/f9/Wolf_building_side.jpg *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Norleaf at English Wikibooks
- **File:Wolf_finished.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/9/9d/Wolf_finished.jpg *License:* GFDL *Contributors:* Transferred from en.wikibooks to Commons. *Original artist:* Norleaf at English Wikibooks
- **File:Wolftut1.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/0/0a/Wolftut1.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks

- **File:Wolftut2.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/3/36/Wolftut2.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Wolftut3.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/9/9a/Wolftut3.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Wolftut4.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b2/Wolftut4.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Wolftut5.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/1/1b/Wolftut5.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Wolftut6.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/b/b2/Wolftut6.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Wolftut7.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/a/ac/Wolftut7.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:Wolftut8.png** *Source:* <https://upload.wikimedia.org/wikipedia/commons/d/da/Wolftut8.png> *License:* GPL *Contributors:* Transferred from en.wikibooks to Commons by Adrignola using CommonsHelper. *Original artist:* The original uploader was AverieZen at English Wikibooks
- **File:World_star_settings_1.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/09/World_star_settings_1.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:World_star_settings_2.jpg** *Source:* https://upload.wikimedia.org/wikipedia/commons/0/06/World_star_settings_2.jpg *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* Animajosser
- **File:Wyvern_--_aaaaaaah!.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/c/c2/Wyvern_--_aaaaaaah%21.png *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* LightspeedLife
- **File:Wyvern_--_aaah!.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/b/b9/Wyvern_--_aaah%21.png *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* LightspeedLife
- **File:Wyvern_in_dev.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/3/3f/Wyvern_in_dev.png *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* LightspeedLife
- **File:Wyvern_teeeeeeeth.png** *Source:* https://upload.wikimedia.org/wikipedia/commons/f/fe/Wyvern_teeeeeeeth.png *License:* CC BY-SA 4.0 *Contributors:* Own work *Original artist:* LightspeedLife

6.3 Content license

- Creative Commons Attribution-Share Alike 3.0