3D Wed Gallery for Game

Title: 3D Web Gallery for Game

By: Ali Akbary

Chapter 4 INTRODUCTION TO APPEARANCE

Learning Outcome

Objectives of this chapter are: -

- Appearance
  - Material
  - Texture

MATERIAL

Materials control the appearance of meshes, curves, volumes and other objects. They define the substance that the object is made of, its color and texture, and how light interacts with it.

Physically based materials can be created using the Principled BSDF, Principled Hair, and Principled Volume shaders. With these uber shaders, a wide variety of materials including plastic, glass, metal, cloth, skin, hair, smoke and fire can be created.

A flexible shading nodes system is used to set up textures and create entirely different types of materials like toon shading.

Setting up Materials

Materials can be created in either the Material properties, or in the Shader Editor. These provide a different view of the same shader nodes and material settings.

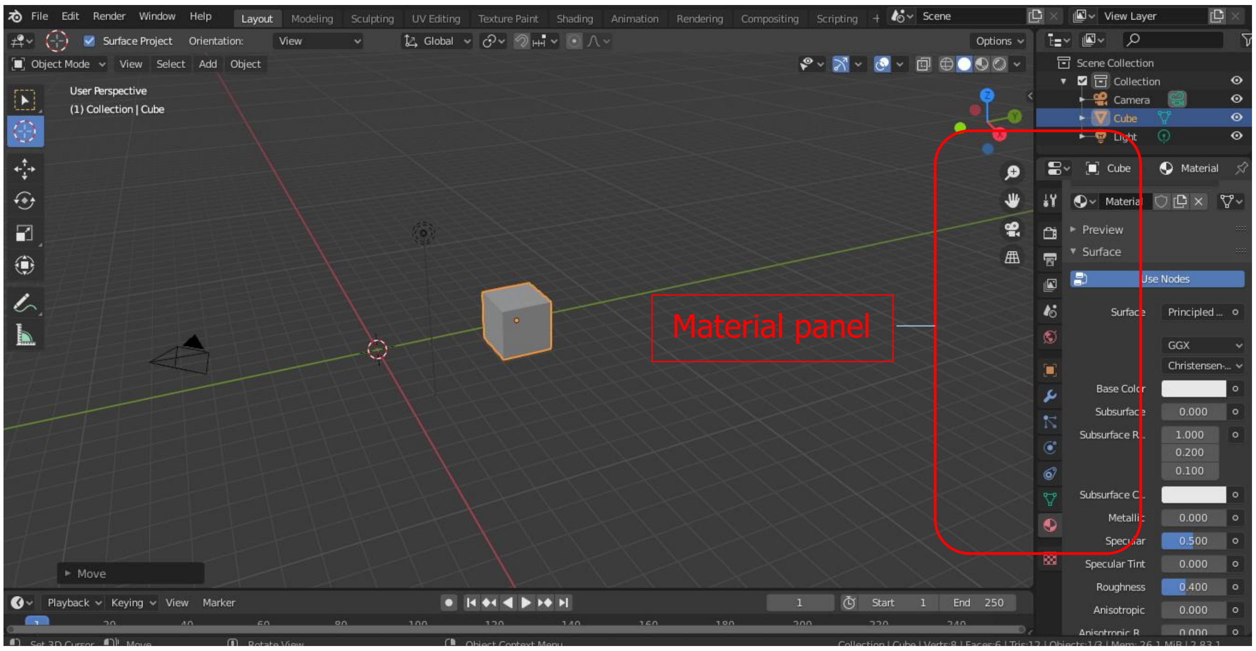
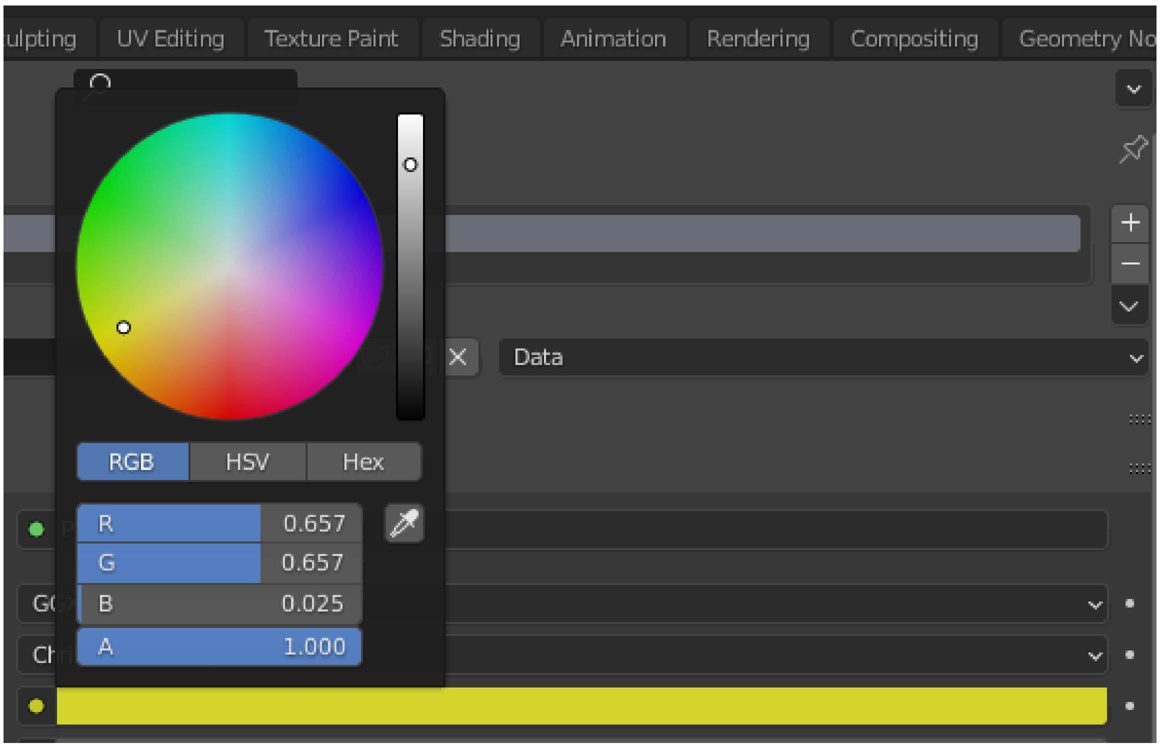


Figure 1 material Properties panel

Choosing material color

We can assign color to object using 3 methods.

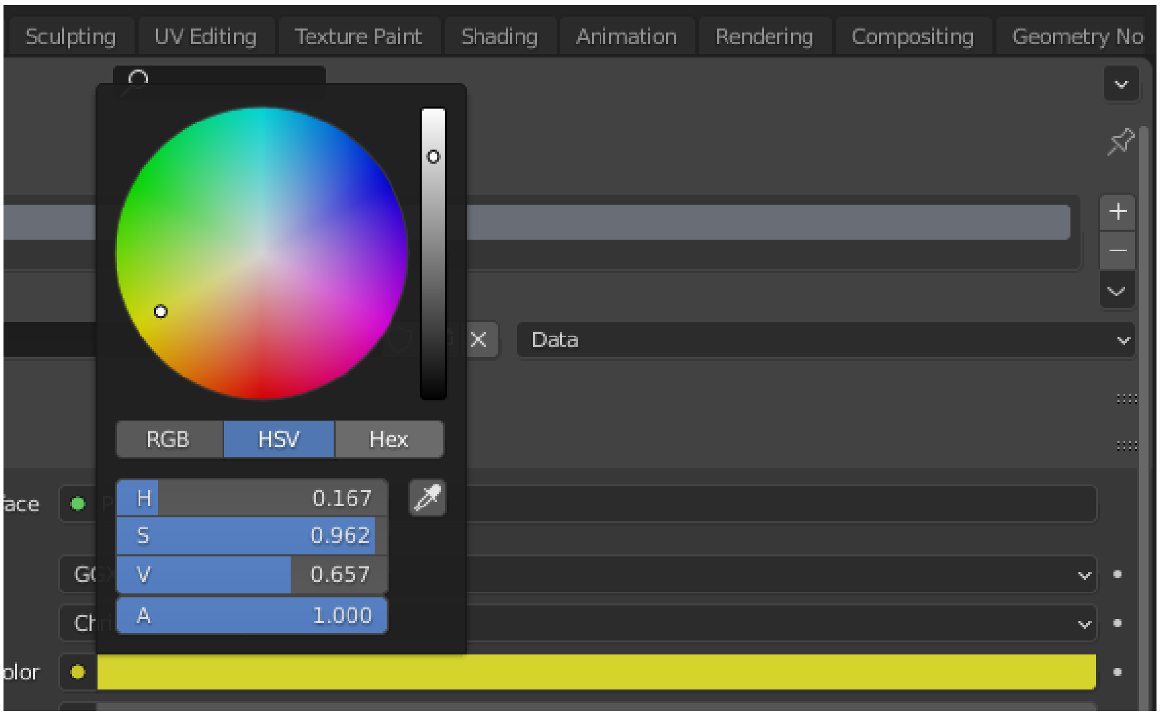
- RGB – Red – Green – Blue



R 0.657
G 0.657
B 0.025
A 1.000

Figure 2 RGB Color

- HSV – Hue – Saturation – Value



H 0.167
S 0.962
V 0.657
A 1.000

Figure 3 HSV Color

➢ HEX – Hexadecimal - Hex is a 6-digit, 24-bit, hexadecimal number that represents Red, Green, and Blue. An example of a Hex color representation is #123456, 12 is Red, 34 is Green, and 56 is Blue. There are 16 million possible color.

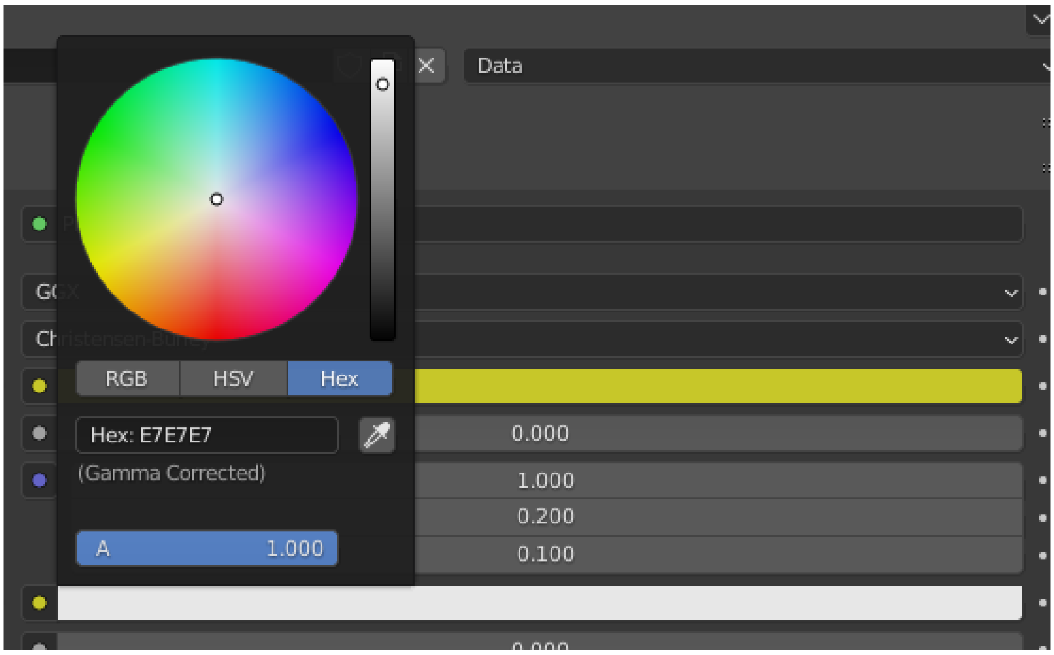
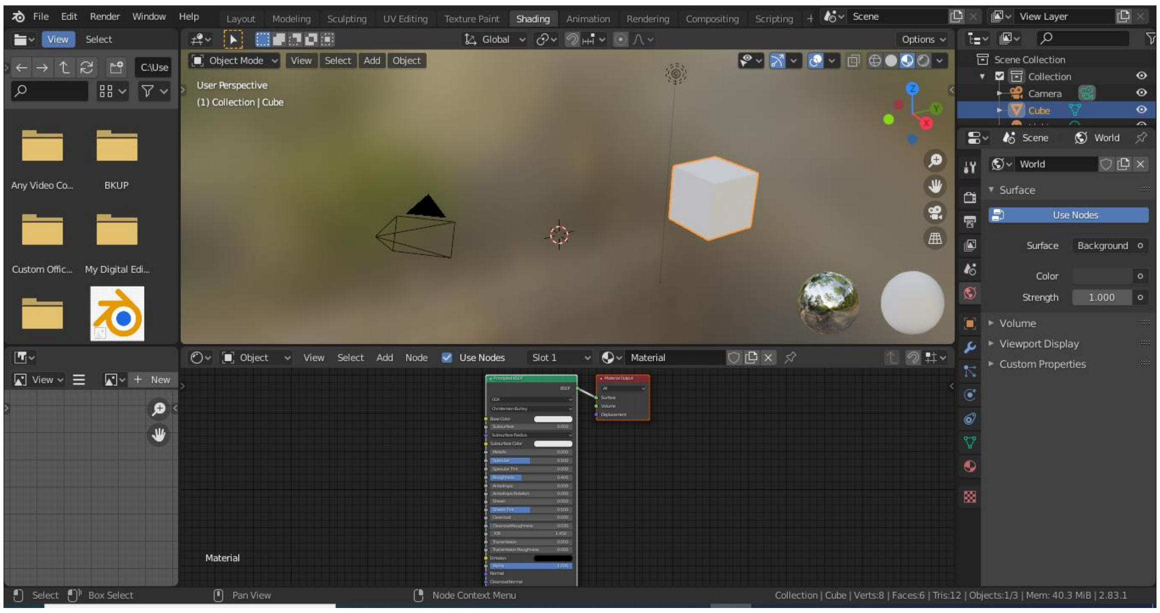


Figure 4 Hexadecimal Color

The default Shading workspace has a Shader Editor and a 3D Viewport that can be set to Material Preview or Rendered shading, to interactively preview how the material interacts with objects and lights in the scene.



Collection | Cube | Verts:8 | Faces:6 | Tris:12 | Objects:1/3 | Mem: 40.3 MiB | 2.83.1

Figure 5 Shader Editor workspace

Materials are data-blocks that can be assigned to one or more objects, and different materials can be assigned to different parts of meshes.

Image textures can be created from scratch in Texture Paint Mode, or by loading in existing images with the Image Texture node. A variety of procedural texture nodes is also available.

Components

Materials consist of three shaders, defining the appearance of the surface, the volume inside the object, and the displacement of the surface.

- Surface shader
- Volume shader
- displacement

Surface Shader

The surface shader controls the textures and light interaction at the surface of the mesh.

Surfaces

The surface shader defines the light interaction at the surface of the mesh. One or more BSDFs specify if incoming light is reflected back, refracted into the mesh, or absorbed.

Emission

Emission defines how light is emitted from the surface, allowing any surface to become a light source.

BSDF

Stands for Bidirectional Scattering Distribution Function. It defines how light is reflected and refracted at a surface.

Reflection

BSDFs reflect an incoming ray on the same side of the surface.

Transmission

BSDFs transmit an incoming ray through the surface, leaving on the other side.

Refraction

BSDFs are a type of Transmission, transmitting an incoming ray and changing its direction as it exists on the other side of the surface.

BSDF Parameters

A major difference from non-physically-based renderers is that direct light reflection from lights and indirect light reflection of other surfaces are not decoupled, but rather handled using a single BSDF. This limits the possibilities a bit, but we believe overall it is helpful in creating consistent-looking renders with fewer parameters to tune.

Roughness

For the glossy BSDFs, the roughness parameter controls the sharpness of the reflection, from 0.0 (perfectly sharp) to 1.0 (very soft).

Volume Shader

The volume shader defines the interior of the mesh. A material can have just a volume shader for cases like smoke and fire, or it can be combined with a surface shader for materials like cloudy glass.

Volume rendering is used to render various effects that cannot be represented by hard surfaces alone.

- Smoke, fire or clouds are set up using a volume object or fluid simulation, with only a volume shader.
- Meshes can also be used to create such shapes by removing the default surface shader and using a volume shader with the mesh shape defining the volume bounds and textures defining the volume density.
- Mist is created with a volume shader for the world, or with a large mesh object encompassing the scene.
- Absorption in glass is simulated by combining a glass surface shader with refraction and a volume absorption shader for the interior of the object.

Shading

Principled Volume

Principled Volume is a physically-based volume shader that can be used to create a wide range of volume materials. It supports scattering, absorption and emission in one easy to use node. Fire can be rendered with blackbody emission.

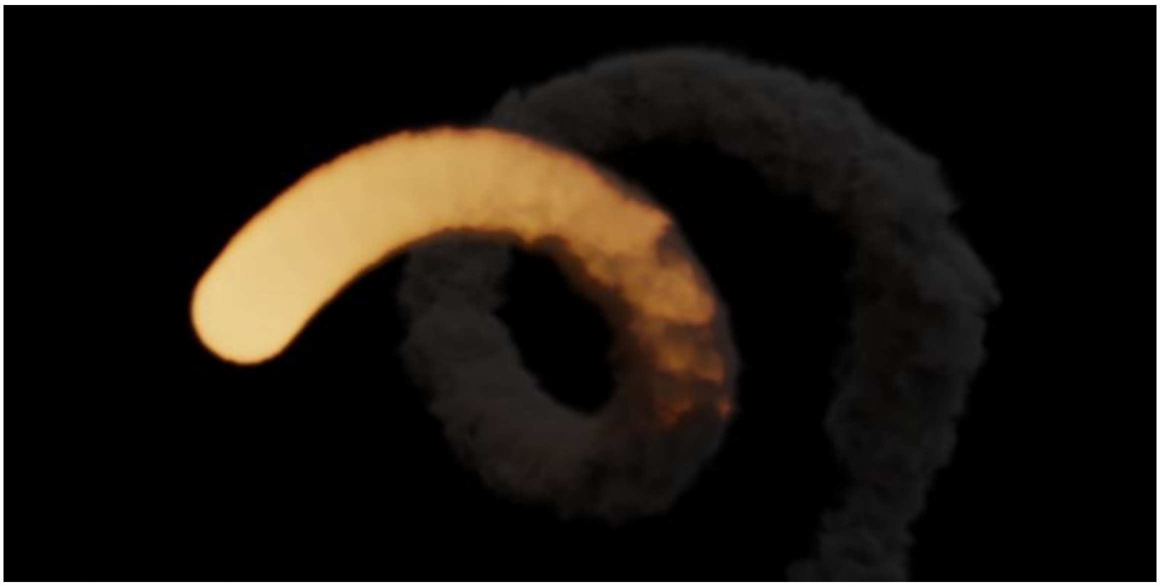


Figure 6 Smoke and fire rendered with Principled Volume shader.

Volume Components

For more control, volume shading components can be manually combined into a custom shader setup.

- Volume Absorption will absorb part of the light as it passes through the volume. This can be used to shade for example black smoke or colored glass objects, or mixed with the Volume Scatter node. This node is similar to the transparent BSDF node, it blocks part of the light and lets other light pass straight through.
- Volume Scatter lets light scatter in other directions as it hits particles in the volume. The anisotropy defines in which direction the light is more likely to scatter. A value of 0 will let light scatter evenly in all directions (similar to the diffuse BSDF node), negative values let light scatter mostly backwards, and positive values let light scatter mostly forward. This can be used to shade white smoke or clouds for example.
- Emission will emit light from the volume, for example for fire.

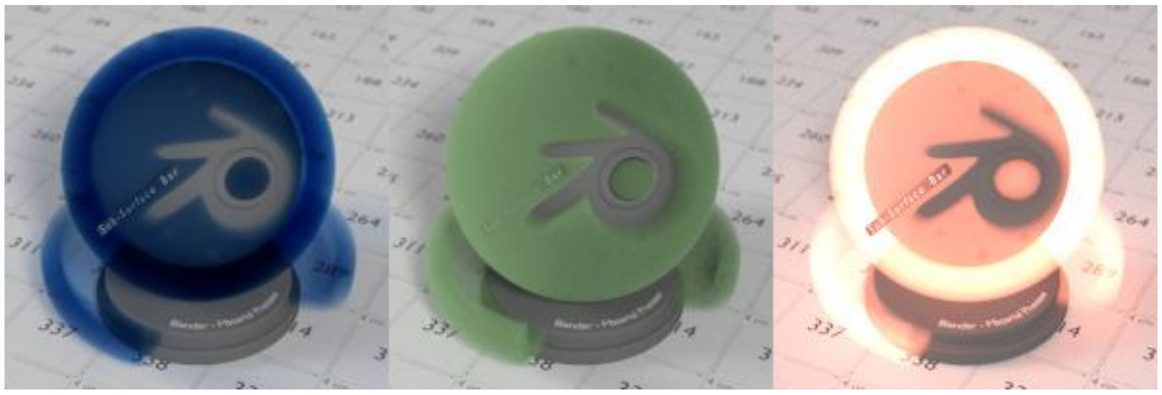


Figure 7 Volume Absorption, Scatter and Emission

Attributes

When rendering smoke and fire, volume attributes are used to define the shape and shading of the volume. The Principled Volume shader will use them by default, while custom volume shaders can use the Attribute node to get attributes such as density, color and temperature.

Density

All volume shaders have a density input. The density defines how much of the light will interact with the volume, getting absorbed or scattered, and how much will pass straight through. For effects such as smoke you would specify a density field to indicate where in the volume there is smoke and how much (density bigger than 0), and where there is no smoke (density equals 0).

Volumes in the real world consist of particles, a higher density means there are more particles per unit volume. More particles means there is a higher chance for light to collide with a particle and get absorbed or scattered, rather than passing straight through.

Mesh Volumes

Meshes used for volume render should be closed and manifold. That means that there should be no holes in the mesh. Each edge must be connected to exactly two faces such that there are no holes or T-shaped faces where three or more faces are connected to an edge.

Normals must point outside for correct results. The normals are used to determine if a ray enters or exits a volume, and if they point in a wrong direction, or there is a hole in the mesh, then the renderer is unable to decide what is the inside or outside of the volume.

These rules are the same as for rendering glass refraction correctly.

World Volume

A volume shader can also be applied to the world, filling the entire space.

Currently, this is most useful for night time or other dark scenes, as the world surface shader or sun lights will have no effect if a volume shader is used. This is because the world background is assumed to be infinitely far away, which is accurate enough for the sun for example. However, for modelling effects such as fog or atmospheric scattering, it is not a good assumption that the volume fills the entire space, as most of the distance between the sun and the earth is empty space. For such effects it is be better to create a volume object surrounding the scene. The size of this object will determine how much light is scattered or absorbed.

Multiple Scattering

Real-world effects such as scattering in clouds or subsurface scattering require many scattering bounces. However, unbiased rendering of such effects can be noisy, so by default the number of bounces is zero in Cycles, and no support is available in Eevee. The effect you get when rendering with zero volume bounces is what is known as “single scattering”, the effect from more bounces is “multiple scattering”.

For rendering materials like skin or milk that require multiple scattering, subsurface scattering is more efficient and easier to control. Particularly the random walk method can accurately render such materials.

For materials such as clouds or smoke that do not have a well-defined surface, volume rendering is required. These look best with many scattering bounces, but in practice one might have to limit the number of bounces to keep render times acceptable.

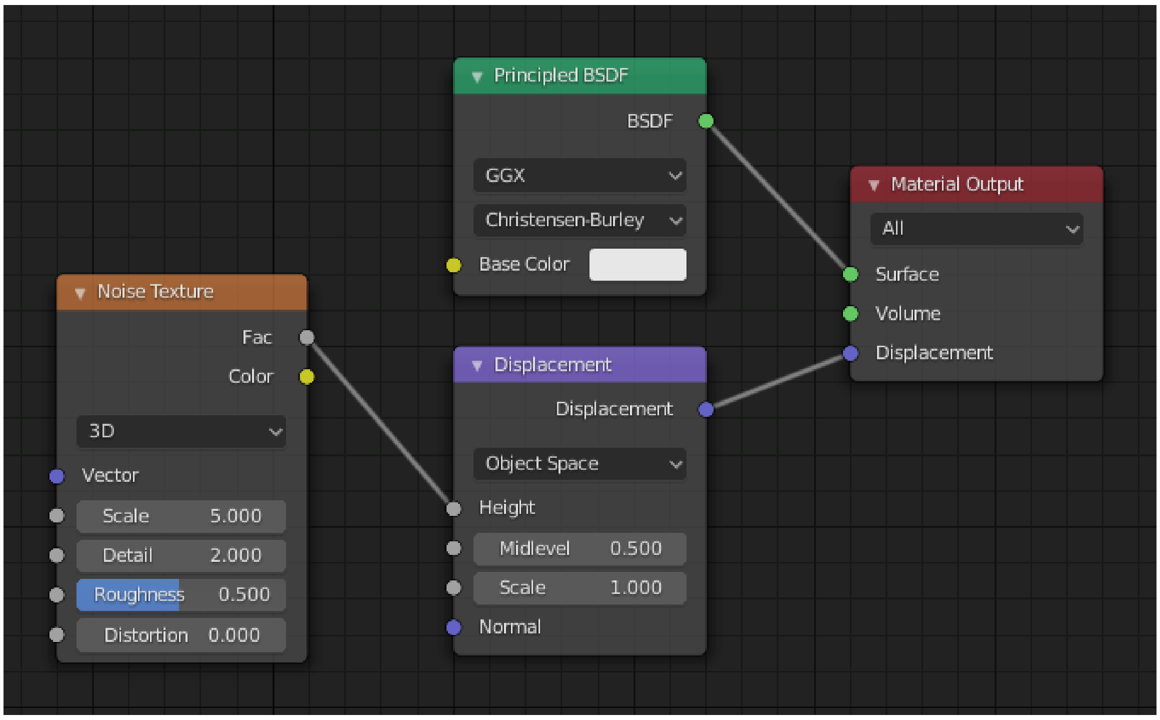
Displacement

The shape of the surface and the volume inside it may be altered by displacement. This way, textures can then be used to make the mesh surface more detailed.

Depending on the settings, the displacement may be virtual, only modifying the surface normals to give the impression of displacement, which is known as bump mapping, or a combination of real and virtual displacement.

Detail can be added to the shape of a surface with displacement shaders.

To create displacement, connect a Displacement or Vector Displacement node to the displacement input of the Material Output node. Procedural, painted or baked textures can then be connected to these nodes.


Principled BSDF
BSDF
GGX
Christensen-Burley
Base Color

Material Output
All
Surface
Volume
Displacement

Noise Texture
Fac
Color
3D
Vector
Scale 5.000
Detail 2.000
Roughness 0.500
Distortion 0.000

Displacement
Displacement
Object Space
Height
Midlevel 0.500
Scale 1.000
Normal

Figure 8 Typical displacement node setup.

Three displacement methods exist, with varying accuracy, performance and memory usage.

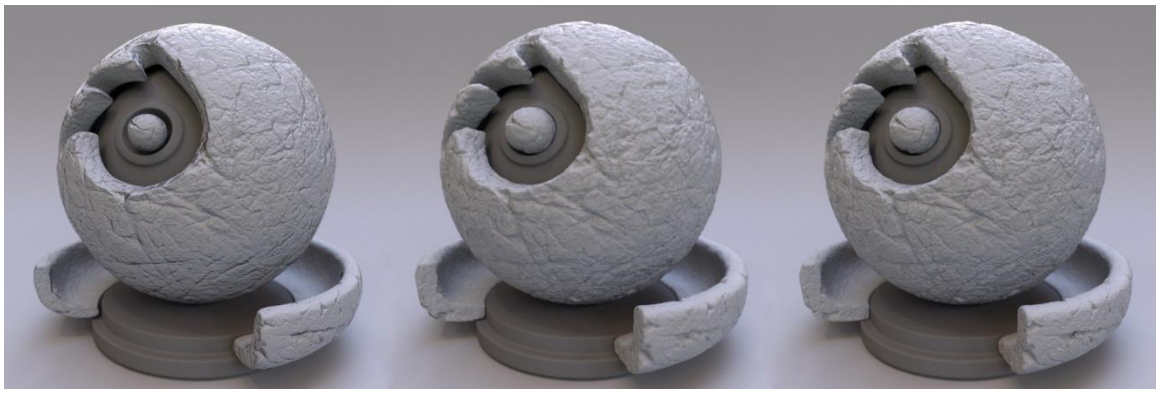


Figure 9 Bump only, displacement only and displacement and bump combined.

Bump Only

The least accurate but most memory efficient method is bump mapping. This method does not actually alter the mesh surface, but merely changes the shading to make it seem so.

Bump maps are often used to add smaller details on a model, for example pores or wrinkles on skin.

For baked bump maps 8-bit images are commonly used, however 16 or 32-bit float maps can provide better looking results. When using image textures use Cubic interpolation to avoid stepping artifacts, these are more visible for bump maps than other types of textures.

Important

Because bump mapping is a fake effect, it can cause artifacts if the actual shape of the geometry is too different from the bump mapped shape. If this happens the strength of bump mapping should be reduced or actual displacement should be used.

Displacement Only

Cycles Only

The most accurate and memory intensive displacement method is to apply true displacement to the mesh surface.

It requires the mesh to be finely subdivided, which can be memory intensive. Adaptive Subdivision is the best way to subdivide the mesh, so that exactly the right amount of subdivision is used depending on the distance of the object to the camera.

For baked displacement maps, best results are achieved with 16 or 32-bit float maps, as 8-bit images often can not represent all the necessary detail.

Displacement and Bump

Cycles Only

Both methods can be combined to use actual displacement for the bigger displacement and bump for the finer details. This can provide a good balance to reduce memory usage.

Once you subdivide the mesh very finely, it is better to use only actual displacement. Keeping bump maps will then only increase memory usage and slow down renders.

Physically Based Shading

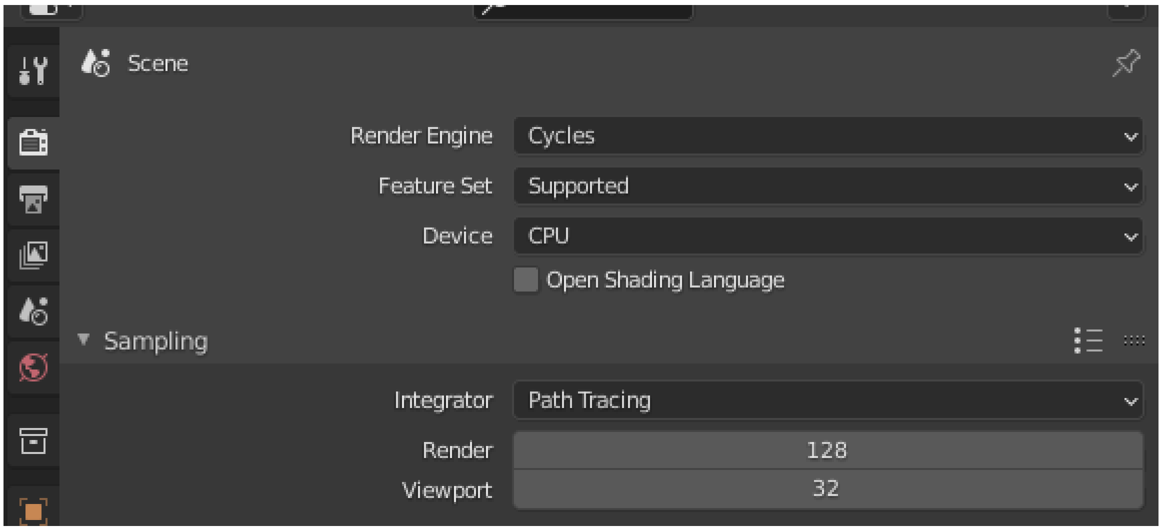
The material system is built with physically-based rendering in mind, separating how a material looks and which rendering algorithm is used to render it. This makes it easier to achieve realistic results and balanced lighting, though there are a few things to keep in mind.

In order for materials to work well with global illumination, they should be energy conserving. That means they cannot reflect more light than comes in. This property is not strictly enforced, but if color are in the range 0.0 to 1.0, and BSDFs are only mixed together with the Mix Shader node, this will automatically be true.

It is however, possible to break this, with color values higher than 1.0 or using the Add Shader node, but one must be careful when doing this to keep materials behaving predictably under various lighting conditions.

Examples of Materials

We can create Mirror, Glass, Metal, Ceramic and others by using materials node. For these experiments we need to change render Engine to "Cycle" in render panel.


Scene

Render Engine: Cycles
Feature Set: Supported
Device: CPU
Open Shading Language

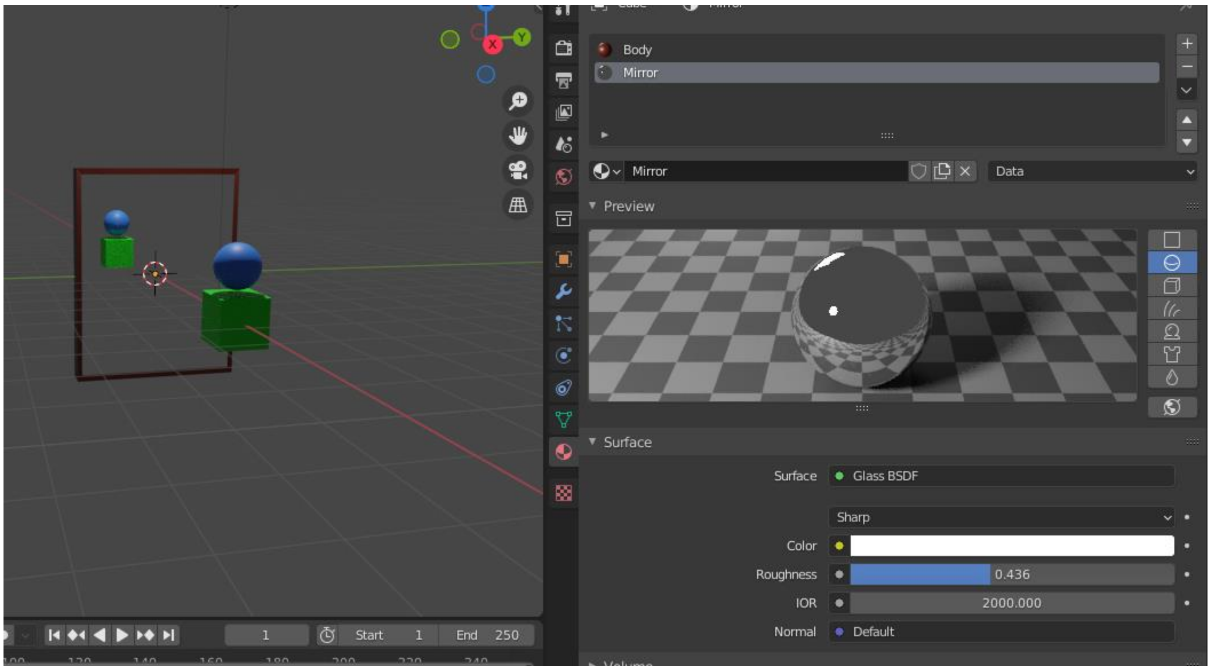
Sampling

Integrator: Path Tracing
Render: 128
Viewport: 32

Figure 10 Selecting Cycle for Render Engine

Mirror

Setting for mirror in the material panel are: -


Body
Mirror

Mirror | Data

Preview

Surface

Surface: Glass BSDF
Sharp
Color
Roughness: 0.436
IOR: 2000.000
Normal: Default

Start 1 End 250

Figure 11 Setting for Mirror

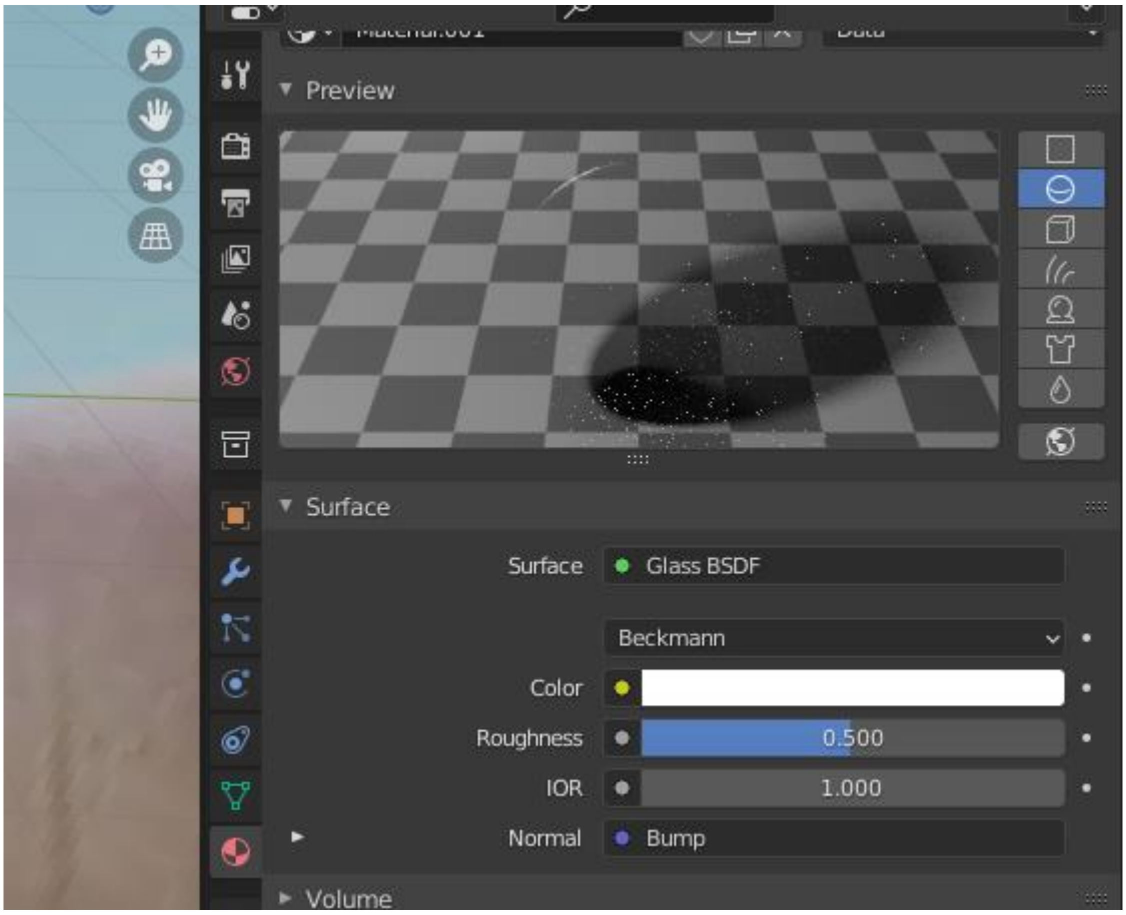
- ➢ Select the face which you want to be mirror
- ➢ Go to material tab click on "+" to add new Material and name Mirror
- ➢ Change the following
    - ❖ Surface – Glass BSDF
    - ❖ Distribution – Sharp
    - ❖ Color – any color
    - ❖ Roughness – 0.5
    - ❖ IOR – more than 1000.000

Glass

In order to convert object to glass we need to change these settings in material panel.

Setting for mirror in the material panel are: -

- Select the face which you want to be Glass
- Go to material tab click on "+" to add new Material and name Glass
- Change the following
    - Surface – Glass BSDF
    - Distribution – Sharp
    - Color – any color
    - Roughness – 0.5
    - IOR – more than 1.000



Surface: Glass BSDF
Beckmann
Color
Roughness: 0.500
IOR: 1.000
Normal: Bump

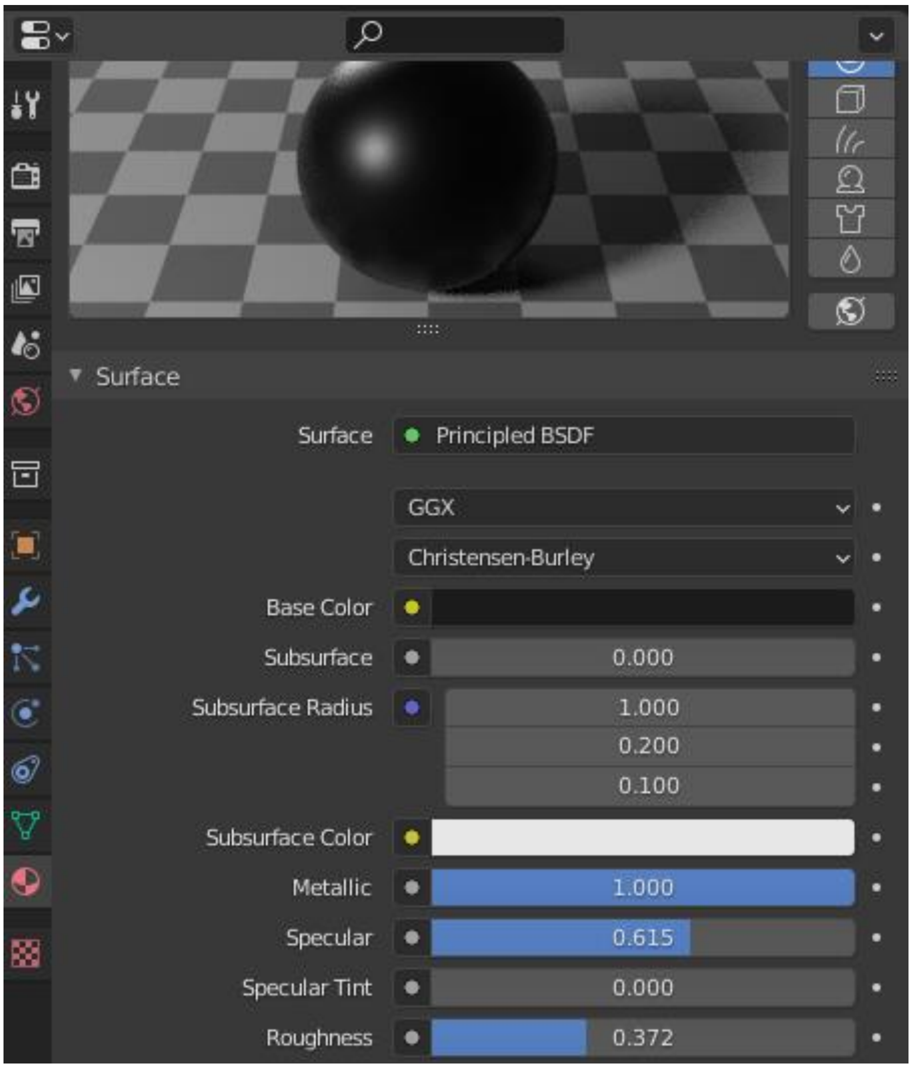
Figure 12 Glass Material Setting

Metal

Setting for mirror in the material panel are: -

- Select the face which you want to be mirror
- Go to material tab click on "+" to add new Material and name Metal
- Change the following
    - Surface – Principled BSDF
    - Color – any color (Tips for more metal color like gold, Silver you can refer to website which provide color codes like adobe color code)

- Roughness – 0.4
- Metallic – 1.000



Surface

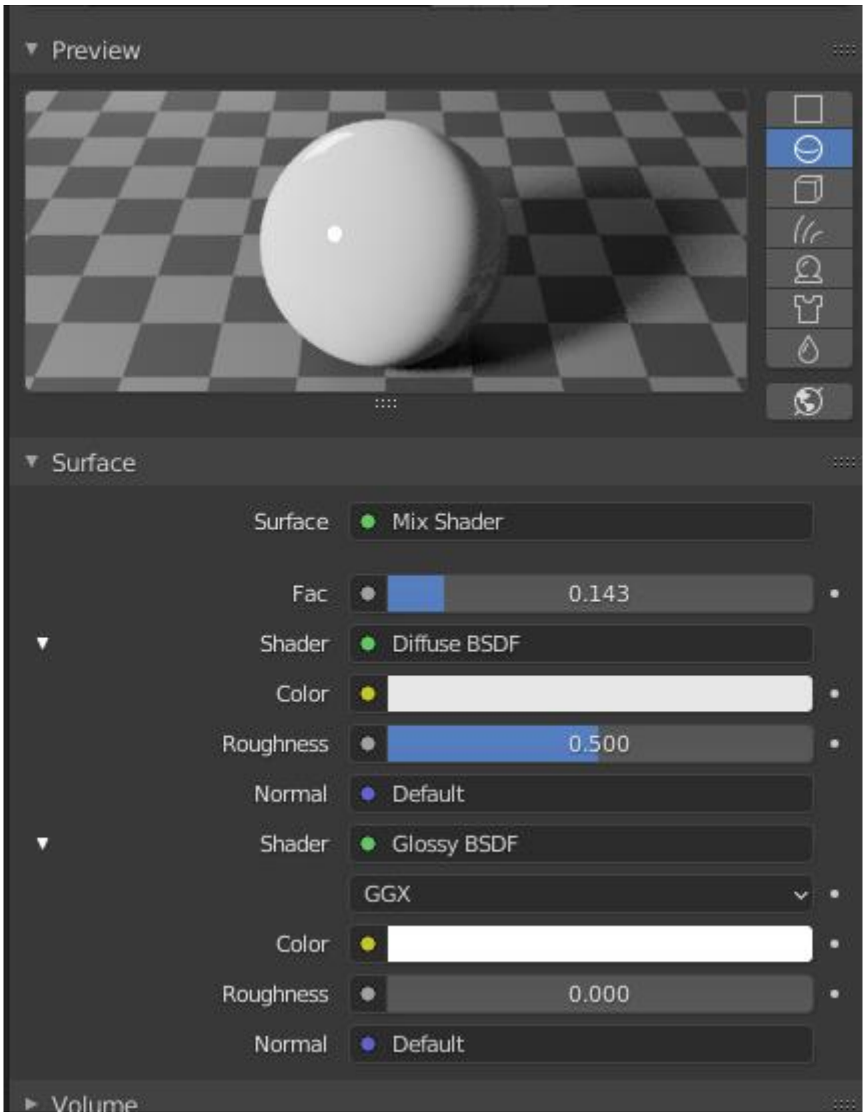
Surface	Principled BSDF
	GGX
	Christensen-Burley
Base Color	
Subsurface	0.000
Subsurface Radius	1.000
	0.200
	0.100
Subsurface Color	
Metallic	1.000
Specular	0.615
Specular Tint	0.000
Roughness	0.372

Figure 13 Metal surface setting example

Ceramic Surface

Setting for Ceramic in the material panel are: -

- Select the face which you want to be Ceramic
- Go to material tab click on "+" to add new Material and name Ceramic
- Change the following
    - Surface – Mixed Shader
        - Fac – 1.0
    - Choose first shader – Diffuse BSDF
    - Choose second Shader – Glossy BSDF
    - Roughness – 0.000



Preview

Surface

Surface	Mix Shader
Fac	0.143
Shader	Diffuse BSDF
Color	
Roughness	0.500
Normal	Default
Shader	Glossy BSDF
	GGX
Color	
Roughness	0.000
Normal	Default

Volume

Figure 14 Ceramic Shader setting

Texturing

The art of giving clothes to the 3D models.

When a 3D model is created, 2D images can be overlaid on it to add colours, designs, and textures. This is called **mapping**, and often the entirety of a model's colour comes from this. These maps can be created in programs like Photoshop, and the illusions of textures can be brushed onto the models as easily as if you painted them yourself; some animators even use real photographs of the textures they're trying to create, simply captured and then altered to make seamless repeatable patterns. This is how many illusions of hair are created; rather than model individual strands, instead of grouped locks of hair are modelled, before a texture is overlaid with individual strands and details painted on.

Introduction

Procedural textures are textures that are defined mathematically. They are generally relatively simple to use, because they do not need to be mapped in a special way. This does not mean that procedural textures cannot become very complex.

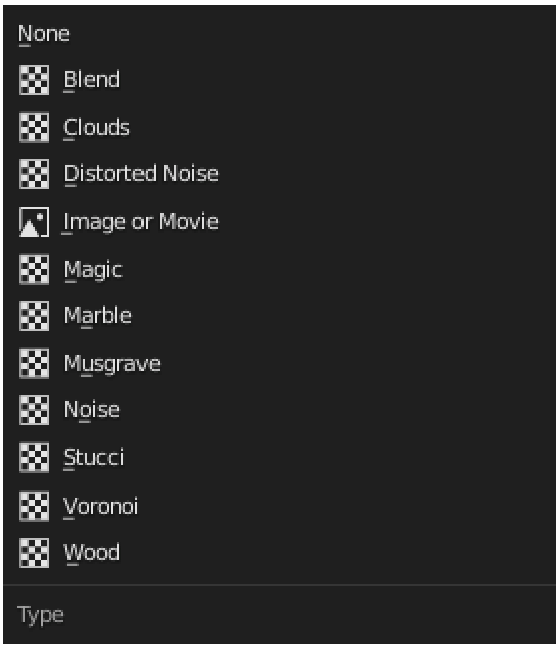


Figure 15 The Texture Type list in the Texture panel of the Texture buttons. (Non-procedural textures darkened out.)

These types of textures are 'real' 3D. By that we mean that they fit together perfectly at the edges and continue to look like what they are meant to look like even when they are cut; as if a block of wood had really been cut in two. Procedural textures are not filtered or anti-aliased. This is hardly ever a problem: the user can easily keep the specified frequencies within acceptable limits.

Noise Basis

Each noise-based Blender texture (except Voronoi and Simple Noise) has a Noise Basis setting that allows the user to select which algorithm is used to generate the texture. This list includes the original Blender noise algorithm. The Noise Basis settings makes the procedural textures extremely flexible (especially Musgrave).

The Noise Basis governs the structural appearance of the texture: -

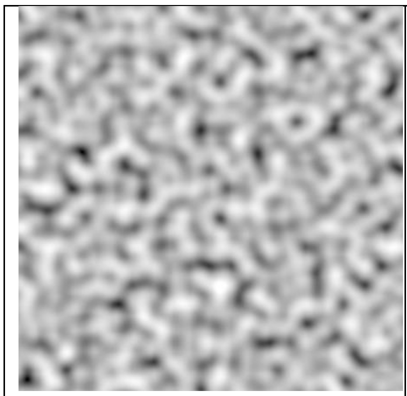
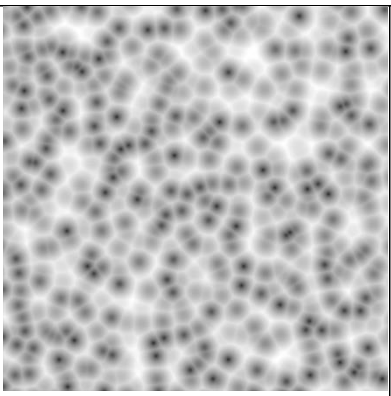
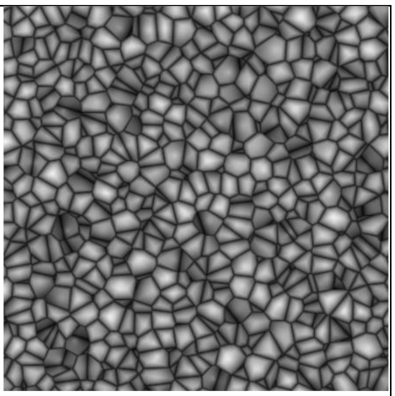
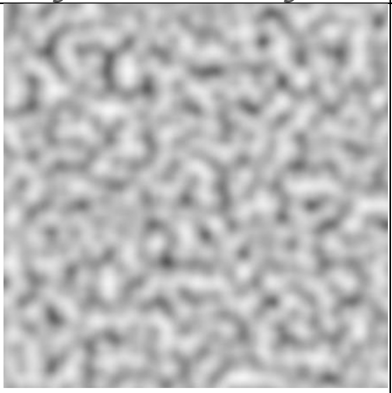
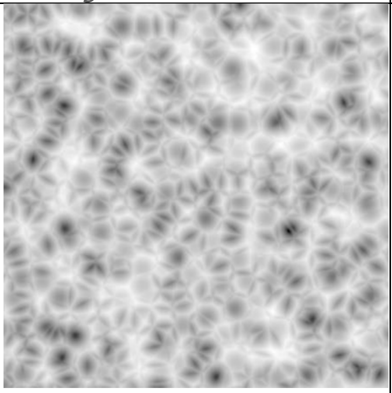
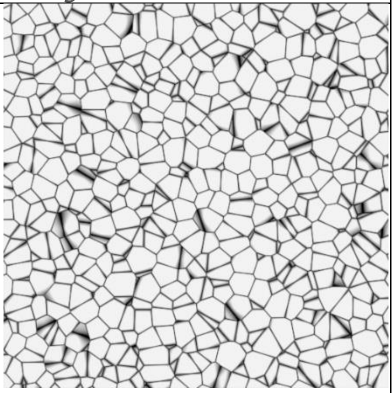
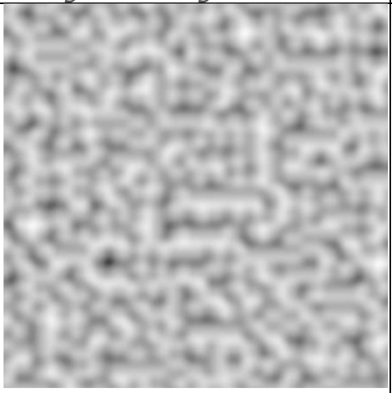
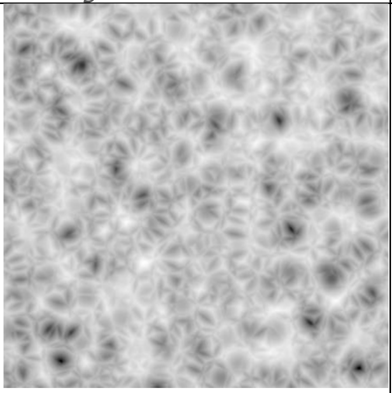
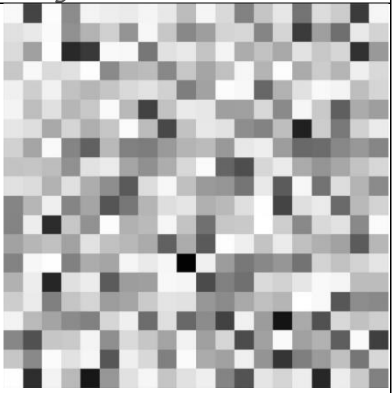
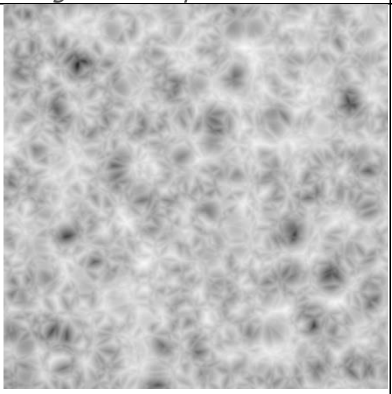
There are two more possible settings for Noise Basis, which are relatively similar to Blender Original: Improved Perlin and Original Perlin.

Nabla

Almost all procedural textures in Blender use derivatives for calculating normals for texture mapping (except Blend and Magic). This is important for Normal and Displacement Maps. The strength of the effect is controlled with the Nabla number field.

Hints

Procedural textures can either produce colored textures, intensity only textures, textures with alpha values and normal textures. If intensity only ones are used the result is a black-and-white texture, which can be greatly enhanced by the use of ramps. If on the other hand you use ramps and need an intensity value, you have to switch on No RGB in the Mapping panel.

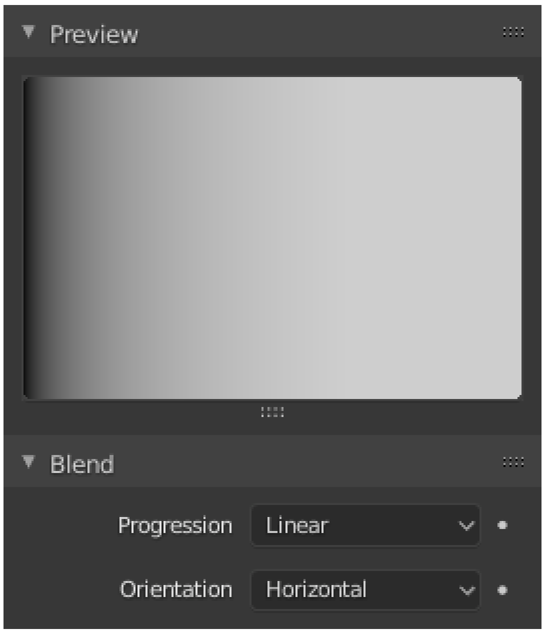
 *Figure 16 Blender Original.*	 *Figure 17 Voronoi F1.*	 *Figure 18 Voronoi F2-F1.*
 *Figure 19 Original Perlin.*	 *Figure 20 Voronoi F2.*	 *Figure 21 Voronoi Crackle.*
 *Figure 22 Improved Perlin.*	 *Figure 23 Voronoi F3.*	 *Figure 24 Cell Noise.*
 *Figure 25 Voronoi F4.*		

Blend

The Blend texture generates a smoothly interpolated progression. This is one of the most frequently used procedural textures. You can use blend textures to blend other textures together (with Stencil), or to create nice effects (especially with the Mapping: Normal trick).

Note

Remember that if you use a ramp to create a custom blending, you may have to use No RGB, if the Mapping value needs an intensity input.



Progression: Linear
Orientation: Horizontal

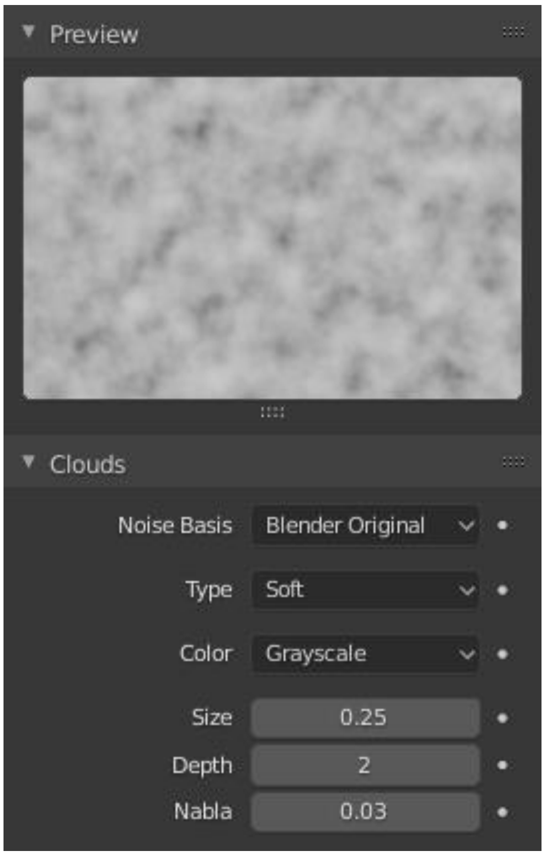
Figure 26 Blend Texture panels.

Options

- Progression
    - **Linear** - A linear progression.
    - **Quadratic** - A quadratic progression.
    - **Easing** - A flowing, nonlinear progression.
    - **Diagonal** - A diagonal progression.
    - **Spherical** - A progression with the shape of a three-dimensional ball.
    - **Quadratic Sphere** - A quadratic progression with the shape of a three-dimensional ball.
    - **Radial** - A radial progression: Horizontal / Vertical. The direction of the progression is flipped a quarter turn.

Clouds

Clouds represent Perlin noise. In addition, each noise-based Blender texture (with the exception of Voronoi and simple noise) has a Noise Basis setting that allows the user to select which algorithm is used to generate the texture. This is often used for Clouds, Fire, Smoke. Well-suited to be used as a Bump map, giving an overall irregularity to the material.



Noise Basis: Blender Original
Type: Soft
Color: Grayscale
Size: 0.25
Depth: 2
Nabla: 0.03

Figure 27 Clouds Texture panels.

Options

- **Grayscale** - The standard noise, gives an intensity.
- **Color** - The noise gives an RGB value.

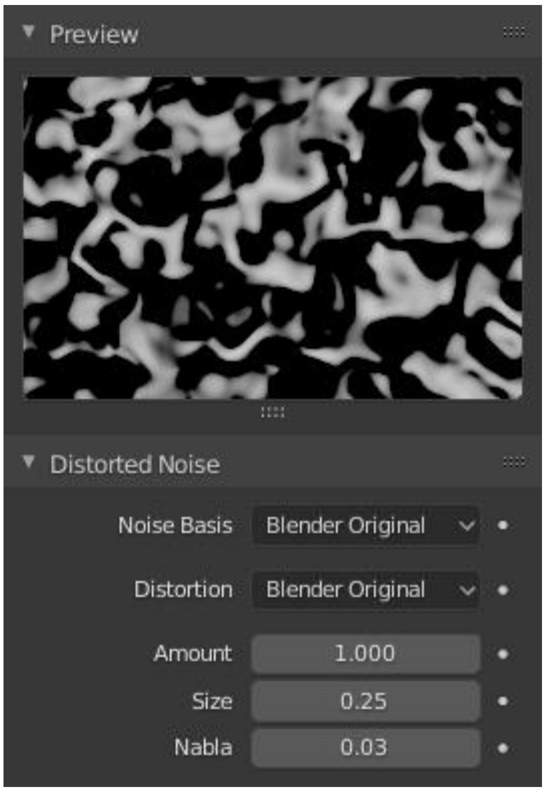
Noise - Soft or Hard, changes contrast and sharpness.

Size - The dimension of the Noise table.

Depth - The depth of the Clouds calculation. A higher number results in a long calculation time, but also in finer details.

Distorted Noise

Distortion Noise takes the option that you pick from Noise Basis and filters it, to create hybrid pattern. It is often used for grunge but is also very complex and versatile.



Noise Basis: Blender Original
Distortion: Blender Original
Amount: 1.000
Size: 0.25
Nabla: 0.03

Figure 28 Distorted Noise Texture panels.

Options

Noise Distortion - The texture to use to distort another.

Basis - The texture to be distorted.

Noise - The size of the noise generated.

Distortion - The amount that Distortion Noise affects Basis.

Image or Movie

The term Image Texture simply means that a graphic image, which is a pixel grid composed of R, G, B, and sometimes Alpha values. It is used as the input source to the texture. As with other types of textures, this information can be used in a number of ways, not only as a simple “decal”.

Video textures are some kind of Image textures and based on movie file or sequence of successive numbered separate images. They are added in the same way that image textures are.

When the Texture Type Image or Movie is selected, three new panels present themselves allowing to control most aspects of how image textures are applied: Image, Image Sampling, and Image Mapping.

About Image-Based Texturing

Texture images take up precious memory space, often being loaded into a special video memory bank that is very fast and very expensive, so it is often very small. So, keep the images as small as possible. A 64×64 image takes up only one fourth the memory of a 128×128 image.

For photorealistic rendering of objects in animations, often larger image textures are used, because the object might be zoomed in on in camera moves. In general, you want to use a texture sized proportionally to the number of pixels that it will occupy in the final render. Ultimately, you only have a certain amount of physical RAM to hold an image texture and the model and to provide workspace when rendering your image.

For the most efficient memory usage, image textures should be square, with dimensions as powers of 2, such as 32×32, 64×64, 128×128, 256×256, 1024×1024, 2048×2048, and 4096×4096.

If you can reuse images across different meshes, this greatly reduces memory requirements. You can reuse images if you map those areas of the meshes that "look alike" to a layout that uses the common image.

When using file textures, it is very important that you have Mapped the UVs of the mesh, and they are laid out appropriately.

You do not have to UV map the entire mesh. The sphere above on the left has some faces mapped, but other faces use procedural materials and textures. Only use UV textures for those portions of your mesh where you want very graphic, precise detail. For example, a model of a vase only needs UV texture for the rim where decorative artwork is incorporated. A throw pillow does not need a different image for the back as the front; in fact many throw pillows have a fabric (procedural material) back.

As another example, you should UV map both eyes of a head to the same image (unless you want one bloodshot and the other clear). Mapping both sides of a face to the same image might not be advisable, because the location of freckles and skin defects are not symmetrical. You could of course change the UV map for one side of the face to slightly offset, but it might be noticeable. Ears are another example where images or section of an image can be mapped to similar faces.

Options

Image - The Image Data-Block Menu.

Alpha

Alpha - Options related to transparency.

Use the alpha channel information stored in the image. Where the alpha value in the image is less than 1.0, the object will be partially transparent and things behind it will be visible. Works with image formats that store transparency information.

Calculate an alpha based on the RGB values of the Image. Black (0, 0, 0) is transparent, white (1, 1, 1) opaque. Enable this option if the image texture is a mask. Note that mask images can use shades of Gray that result in semi-transparency, like ghosts, flames, and smoke/fog.

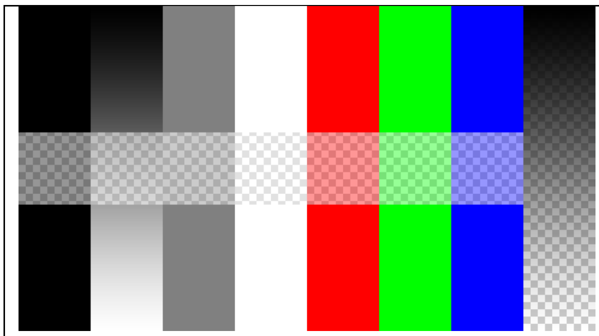


Figure 29 Image with Use alpha. The alpha values of the pixels are evaluated.

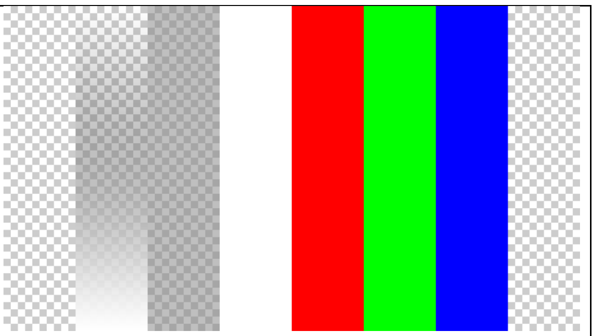


Figure 30 Image with Calculate alpha only, Use Alpha in the Image panel is disabled.

Invert - Reverses the alpha value. Use this option if the mask image has white where you want it transparent and vice versa.

Mapping

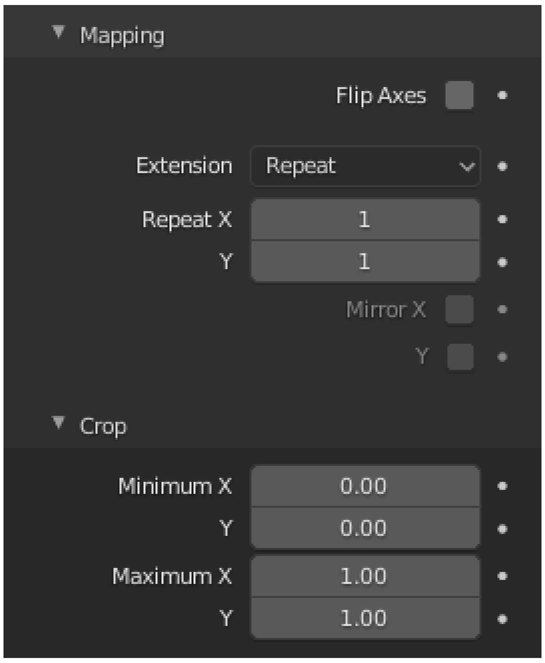

Mapping
Flip Axes
Extension: Repeat
Repeat X: 1
Y: 1
Mirror X
Y
Crop
Minimum X: 0.00
Y: 0.00
Maximum X: 1.00
Y: 1.00

Figure 31 Image Mapping panel.

In the Mapping panel, you can control how the image is mapped or projected onto the 3D model.

Flip X/Y Axis - Rotates the image 90 degrees counter clockwise when rendered.

- **Extension**
  - **Extend** - Outside the image the color of the edges are extended.
  - **Clip** - Clip to image size and set exterior pixels as transparent. Outside the image, an alpha value of 0.0 is returned. This allows you to 'paste' a small logo on a large object.
  - **Clip Cube** - Clips to cubic-shaped area around the images and sets exterior pixels as transparent. The same as Clip, but now the 'Z' coordinate is

calculated as well. An alpha value of 0.0 is returned outside a cube-shaped area around the image.

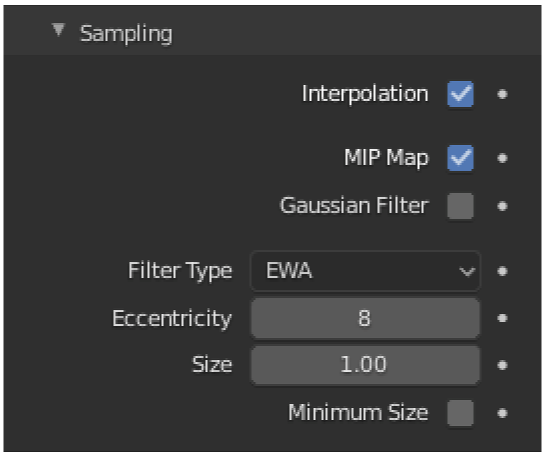
- **Repeat** - The image is repeated horizontally and vertically.
    - Repeat - X/Y repetition multiplier.
    - Mirror - Mirror on X/Y axes. These buttons allow you to map the texture as a mirror, or automatic flip of the image, in the corresponding X and/or Y direction.
    - Checker - Checkerboards quickly made. You can use the option size on the Mapping panel as well to create the desired number of checkers.
        - **Even/Odd** - Set even/odd tiles.
        - **Distance** - Governs the distance between the checkers in parts of the texture size.

- Crop Minimum / Crop Maximum

The offset and the size of the texture in relation to the texture space. Pixels outside this space are ignored. Use these to crop, or choose a portion of a larger image to use as the texture.

Sampling

In the Sampling panel you can control how the information is retrieved from the image.

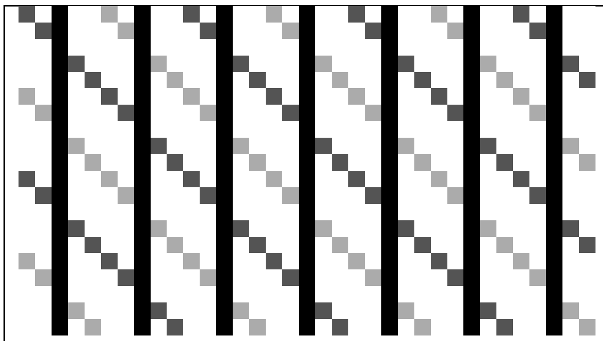
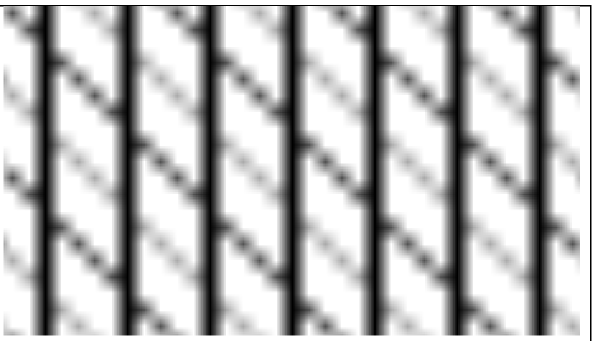


Sampling
Interpolation ✓
MIP Map ✓
Gaussian Filter ☐
Filter Type: EWA
Eccentricity: 8
Size: 1.00
Minimum Size ☐

Figure 32 Image Sampling panel.

Interpolation

This option interpolates the pixels of an image. This becomes visible when you enlarge the picture. By default, this option is on. Turn this option off to keep the individual pixels visible and if they are correctly anti-aliased. This last feature is useful for regular patterns, such as lines and tiles; they remain 'sharp' even when enlarged considerably. Turn this image off if you are using digital photos to preserve crispness.

 *Figure 33 Enlarged Image texture without Interpolation.*	 *Figure 34 Enlarged Image texture with Interpolation.*

MIP Map

Mip-maps are precalculated, smaller, filtered textures for a certain size. A series of pictures is generated, each half the size of the former one. This optimizes the filtering process. By default, this option is enabled and speeds up rendering. When this option is off, you generally get a sharper image, but this can significantly increase calculation time if the filter dimension (see below) becomes large. Without mip-maps you may get varying pictures from slightly different camera angles, when the textures become very small. This would be noticeable in an animation.

MIP Map Gaussian filter

Used in conjunction with mip-mapping, it enables the mip-map to be made smaller based on color similarities. In game engines, you want your textures, especially your mip-map textures, to be as small as possible to increase rendering speed and frame rate.

Filter - The filter size used in rendering, and also by the options Mip Map and Interpolation. If you notice Gray lines or outlines around the textured object, particularly where the image is transparent, turn this value down from 1.0 to 0.1 or so.

Texture Filter Type

Texture filter to use for image sampling. Just like a pixel represents a picture element, a Texel represents a texture element. When a texture (2D texture space) is mapped onto a 3D model (3D model space), different algorithms can be used to compute a value for each pixel based on samples from several Texels.

Box - A fast and simple nearest-neighbour interpolation known as Monte Carlo integration.

EWA (Elliptical Weighted Average) - One of the most efficient direct convolution algorithms developed by Paul Heckbert and Ned Greene in the 1980s. For each texel, EWA samples, weights, and accumulates texels within an elliptical footprint and then divides the result by the sum of the weights.

Eccentricity - Maximum Eccentricity. Higher values give less blur at distant/oblique angles, but is slower.

FELINE (Fast Elliptical Lines) - Uses several isotropic probes at several points along a line in texture space to produce an anisotropic filter to reduce aliasing artifacts without considerably increasing rendering time.

Probes - Number of probes to use. An integer between 1 and 256. Further reading: McCormack, J; Farkas, KI; Perry, R; Jouppi, NP (1999) Simple and Table Feline: Fast Elliptical Lines for Anisotropic Texture Mapping, WRL

Area - Area filter to use for image sampling.

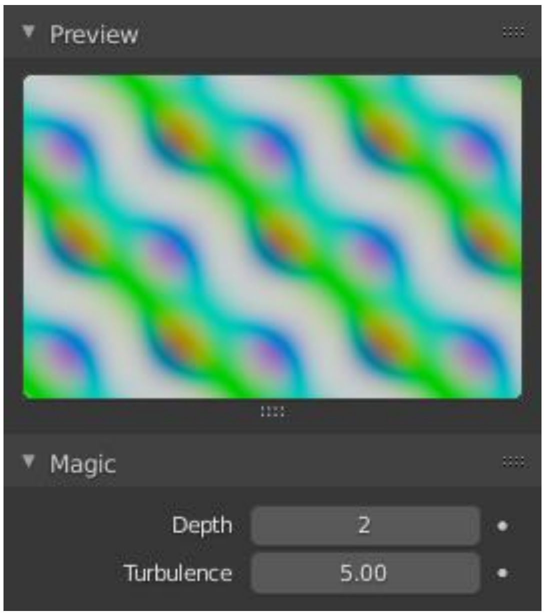
Eccentricity - Maximum Eccentricity. Higher values give less blur at distant/oblique angles, but is slower.

Filter Size - The filter size used by MIP Map and Interpolation.

Minimum Filter Size - Use Filter Size as a minimal filter value in pixels.

Magic

The magic texture is not frequently used. It can be used for "Thin Film Interference", if you set Mapping to Reflection and use a relatively high Turbulence. The RGB components are generated independently with a sine formula.



Depth: 2
Turbulence: 5.00

Figure 35 Magic Texture panels.

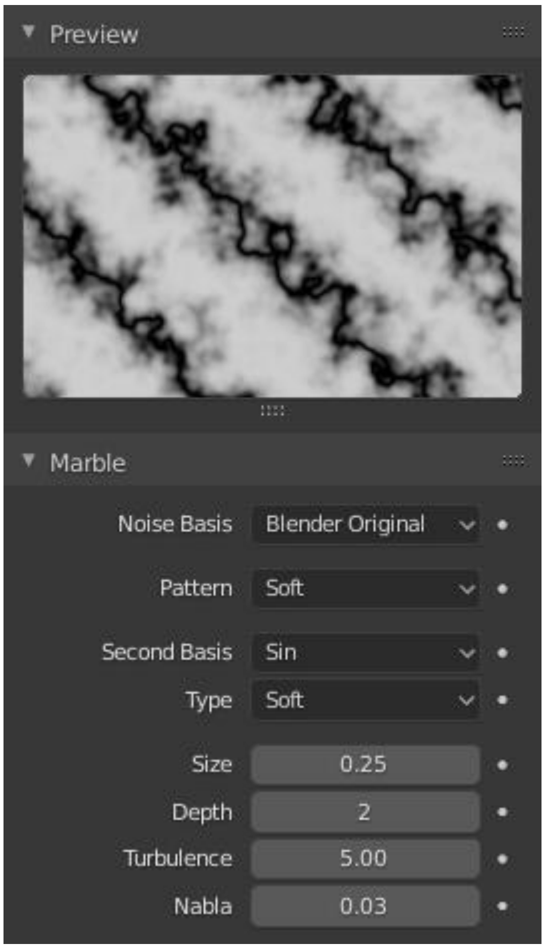
Options

Depth - The depth of the calculation. A higher number results in a long calculation time, but also in finer details.

Turbulence - The strength of the pattern.

Marble

The marble texture is used to generate marble, fire, or noise with a structure. Bands are generated based on the sine, saw, or triangular formula and noise turbulence.


Preview

Marble

Noise Basis: Blender Original

Pattern: Soft

Second Basis: Sin

Type: Soft

Size: 0.25

Depth: 2

Turbulence: 5.00

Nabla: 0.03

Figure 36 Marble Texture panels.

Options

Marble Type - Three settings for soft to more clearly defined Marble. Soft, Sharp, Sharper.

Noise basis - Shape of wave to produce bands. Sine, Saw, Triangle

Noise Type - The noise function works with two methods. Soft, Hard

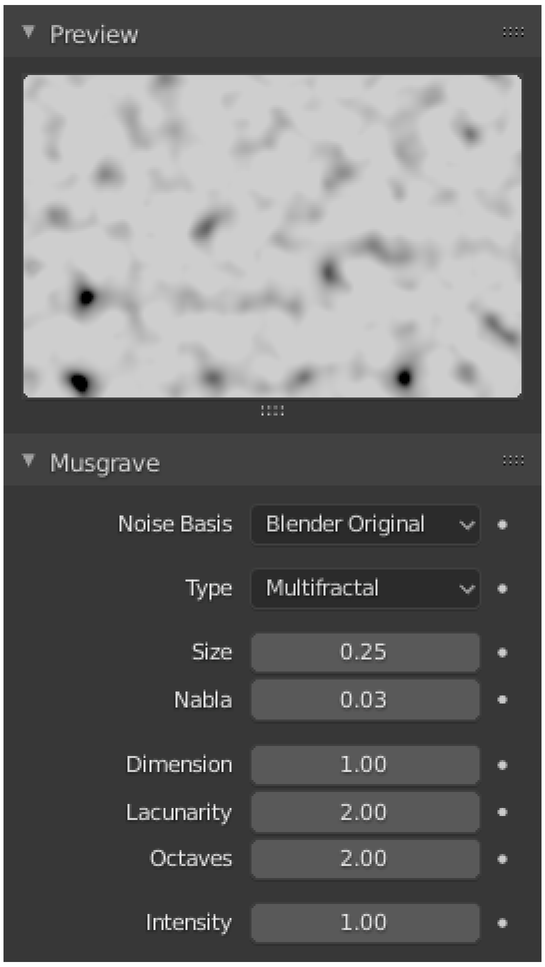
Size - The dimensions of the noise table.

Depth - The depth of the Marble calculation. A higher value results in greater calculation time, but also in finer details.

Turbulence - The turbulence of the sine bands.

Musgrave

The musgrave texture is used to generate organic materials, but it is very flexible. You can do nearly everything with it.


Preview

Musgrave

Noise Basis: Blender Original

Type: Multifractal

Size: 0.25

Nabla: 0.03

Dimension: 1.00

Lacunarity: 2.00

Octaves: 2.00

Intensity: 1.00

Figure 37 Musgrave Texture panels

.

Options

Type

This procedural texture has five noise types on which the resulting pattern can be based and they are selectable from a select menu at the top of the tab. The five types are: -

- Hetero Terrain
- Fractal Brownian Motion (fBm)
- Hybrid Multifractal
- Ridged Multifractal
- Multifractal

These noise types determine the manner in which Blender layers successive copies of the same pattern on top of each other at varying contrasts and scales.

Examples with Basis: Voronoi: F1, Dimension: 0.5, Lacunarity: 0.15, Octave: 2.0.

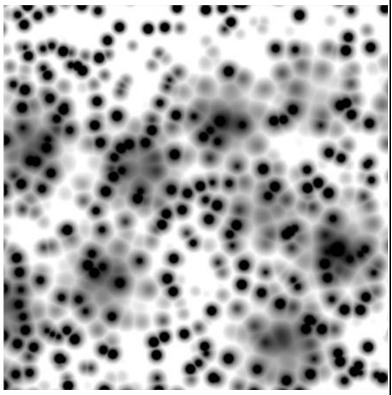


Figure 38 Hetero Terrain.

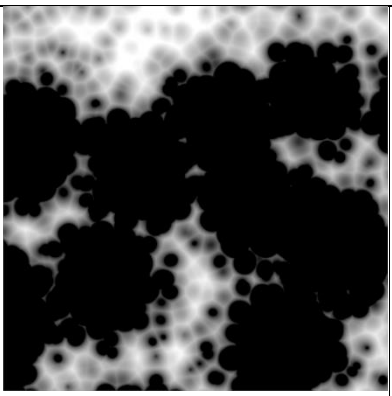


Figure 39 Fractal Brownian Motion.

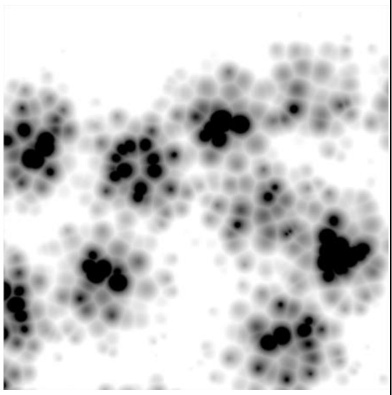


Figure 40 Hybrid Multifractal.

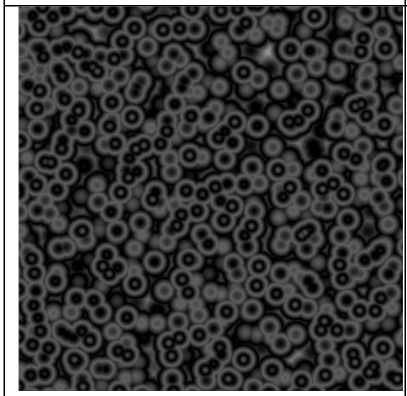


Figure 41 Ridged Multifractal.

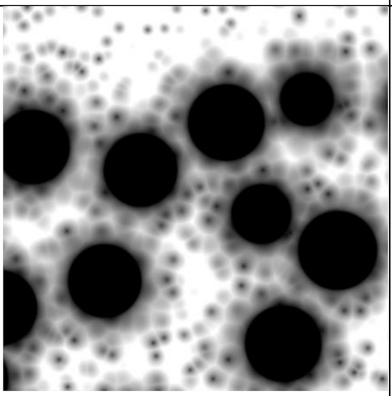


Figure 42 Multifractal.

The main noise types have four characteristics: -

Dimension - Fractal dimension controls the contrast of a layer relative to the previous layer in the texture. The higher the fractal dimension, the higher the contrast between each layer, and thus the more detail shows in the texture.

Lacunarity - Lacunarity controls the scaling of each layer of the Musgrave texture, meaning that each additional layer will have a scale that is the inverse of the value which shows on the button. i.e. Lacunarity = 2 –> Scale = 1/2 original.

Octaves - Octave controls the number of times the original noise pattern is overlayed on itself and scaled/contrasted with the fractal dimension and lacunarity settings.

Intensity - Light intensity. Called Offset for Hetero Terrain.

The Hybrid Multifractal and Ridged Multifractal types have these additional settings:

Offset - Both have a "Fractal Offset" button that serves as a "sea level" adjustment and indicates the base height of the resulting bump map. Bump values below this threshold will be returned as zero.

Gain - Setting which determines the range of values created by the function. The higher the number, the greater the range. This is a fast way to bring out additional details in a texture where extremes are normally clipped off.

Noise

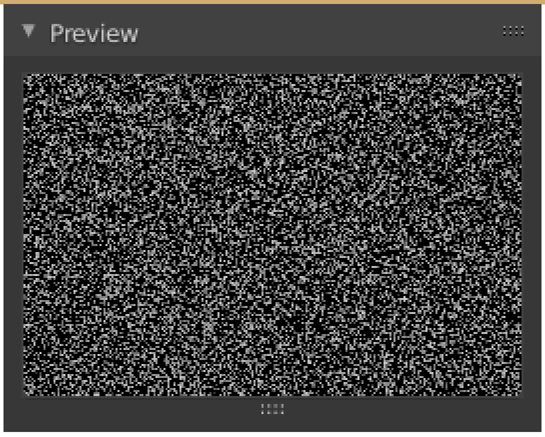


Figure 43 Noise Texture panel.

Although this looks great, it is not Perlin Noise! This is a true, randomly generated Noise. This gives a different result every time, for every frame, for every pixel.

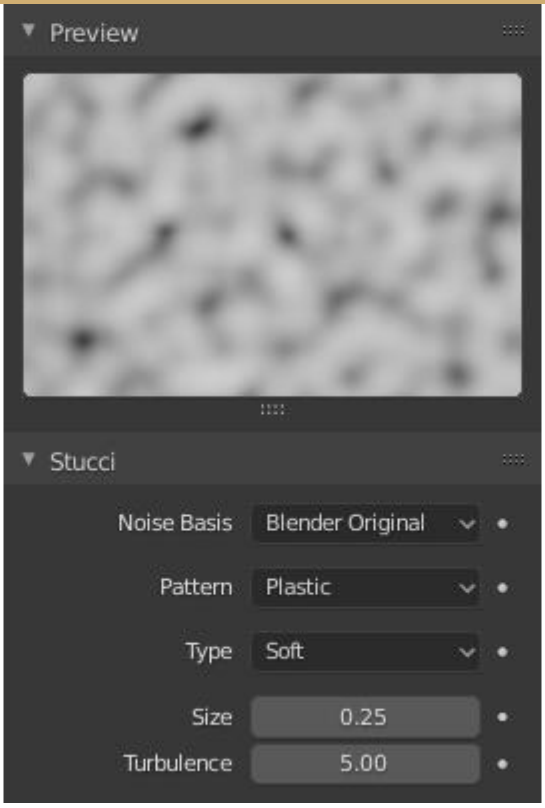
Options

There are no options for this noise.

Often used for

White noise in an animation. This is not well suited if you do not want an animation. For material displacement or bump, use clouds instead.

Stucci



Noise Basis: Blender Original
Pattern: Plastic
Type: Soft
Size: 0.25
Turbulence: 5.00

Figure 44 Stucci Texture panels.

The Stucci texture is based on noise functions. It is often used for stone, asphalt, or oranges, normally for bump mapping to create grainy surfaces.

Options

Plastic / Wall In / Wall out

Plastic is the standard Stucci, while the "walls" is where Stucci gets it name. This is a typical wall structure with holes or bumps.

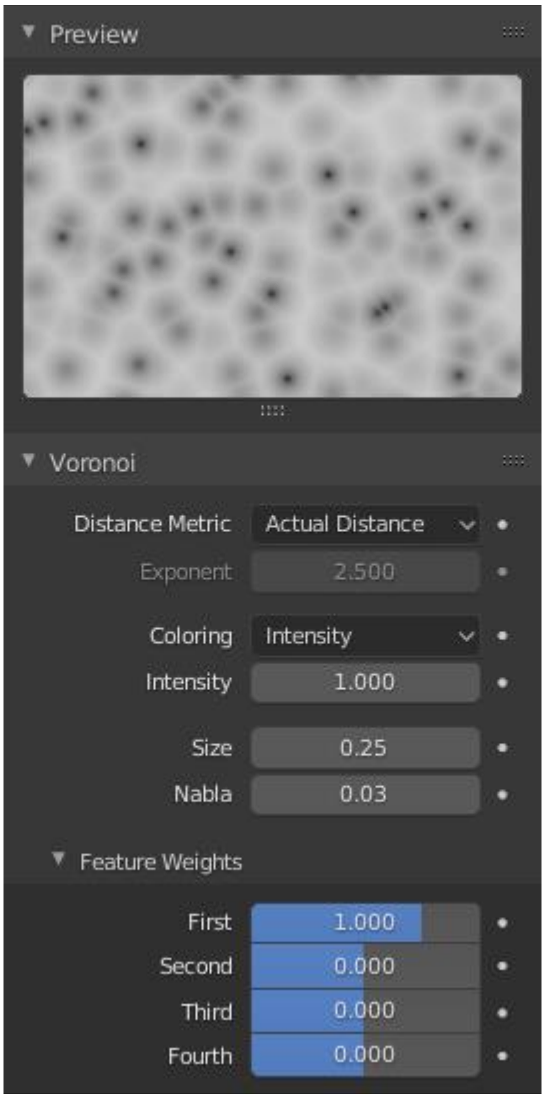
Soft / Hard - There are two methods available for working with Noise.

Size - Dimension of the Noise table.

Turbulence - Depth of the Stucci calculations.

Voronoi

The Voronoi texture is used to generate very convincing Metal, especially the "Hammered" effect. Organic shaders (e.g. scales, veins in skin).



Distance Metric: Actual Distance
Exponent: 2.500
Coloring: Intensity
Intensity: 1.000
Size: 0.25
Nabla: 0.03
First: 1.000
Second: 0.000
Third: 0.000
Fourth: 0.000

Figure 45 Voronoi Texture panels.

Options

Distance Metric

This procedural texture has seven Distance Metric options. These determine the algorithm to find the distance between cells of the texture. These options are:

- Minkowski
- Minkowski 4
- Minkowski 1/2
- Chebychev
- Manhattan
- Distance Squared
- Actual Distance

The Minkowski setting has a user definable value (the Exponent button) which determines the Minkowski exponent e of the distance function:

(xe + ye + ze)1/e

A value of one produces the Manhattan distance metric, a value less than one produces stars (at 0.5, it gives a Minkowski 1/2), and higher values produce square cells (at 4.0, it gives a Minkowski 4, at 10.0, a Chebychev). So nearly all Distance Settings are basically the same – a variation of Minkowski.

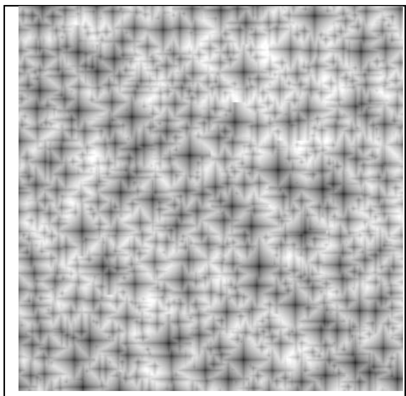
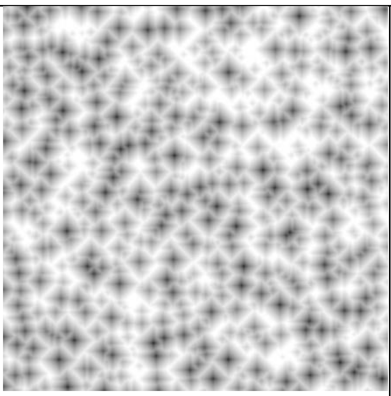
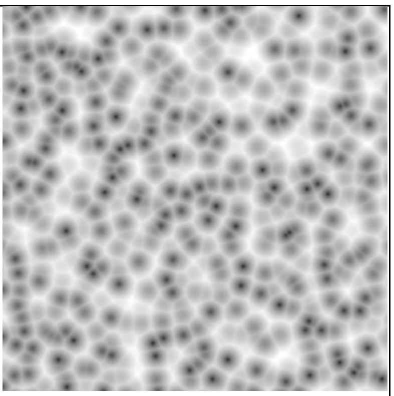
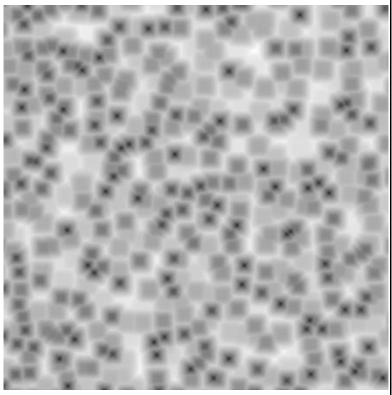
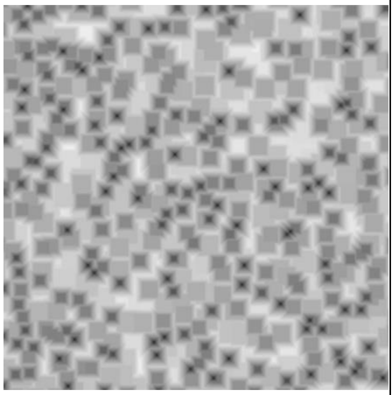
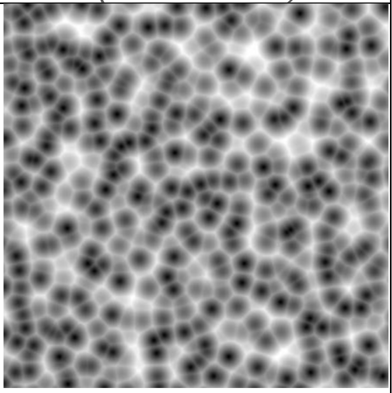
You can get irregularly-shaped rounded cells with the Actual Distance / Distance Squared options.

Feature Weights

These four sliders at the bottom of the Voronoi panel represent the values of the four Worley constants, which are used to calculate the distances between each cell in the texture based on the distance metric. Adjusting these values can have some interesting effects on the end result…

Coloring

Four settings (Intensity, Position, Position and Outline, and Position, Outline, and Intensity) that can use four different noise basis as methods to calculate color and intensity of the texture output. This gives the Voronoi texture you create with the "Worley Sliders" a completely different appearance and is the equivalent of the noise basis setting found on the other textures.

 *Figure 46 Minkowski Exponent: 0.5 (Minkowski 1/2).*	 *Figure 47 Minkowski Exponent: 1 (Manhattan).*	 *Figure 48 Minkowski Exponent: 2 (Actual Distance).*
 *Figure 49 Minkowski Exponent: 4 (Minkowski 4).*	 *Figure 50 Minkowski Exponent: 10 (Chebychev).*	 *Figure 51 Distance Squared (more contrast than Actual Distance).*

Wood

The wood texture is used to generate wood and ring-shaped patterns.

Options

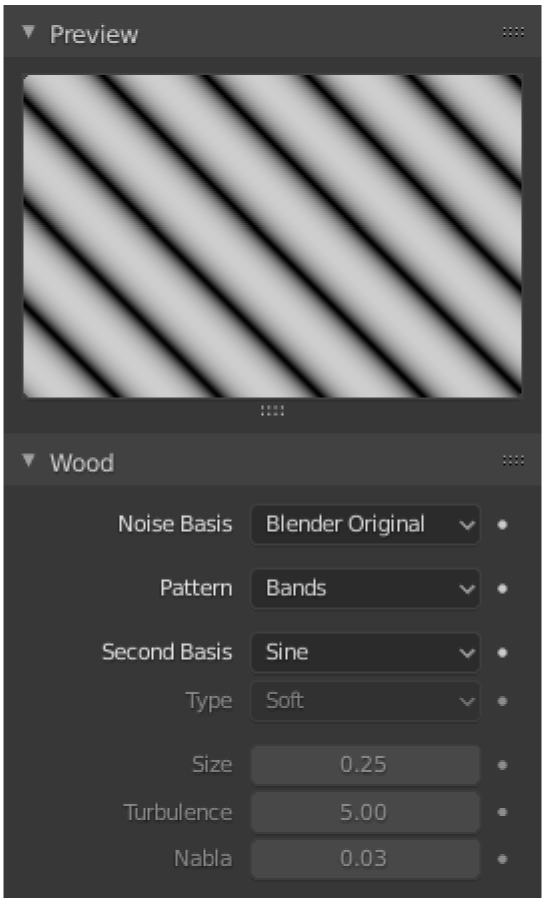
Noise Basis - Shape of wave to produce bands. Sine, Saw, Triangle

Wood Type - Set the bands to either straight or ring-shaped, with or without turbulence. Bands, Rings, Band Noise, Ring Noise

Noise Type - There are two methods available for the Noise function. Soft, Hard

Size - Dimension of the Noise table.

Turbulence - Turbulence of the Band Noise and Ring Noise types.


Preview

Wood

Noise Basis: Blender Original

Pattern: Bands

Second Basis: Sine

Type: Soft

Size: 0.25

Turbulence: 5.00

Nabla: 0.03

Figure 52 Wood Texture panels.