



Design Tools for Satellite Models

Daniel Topa
daniel.topa@hii-tsd.com

Huntington Ingalls Industries
Mission Technologies

November 14, 2024



Overview

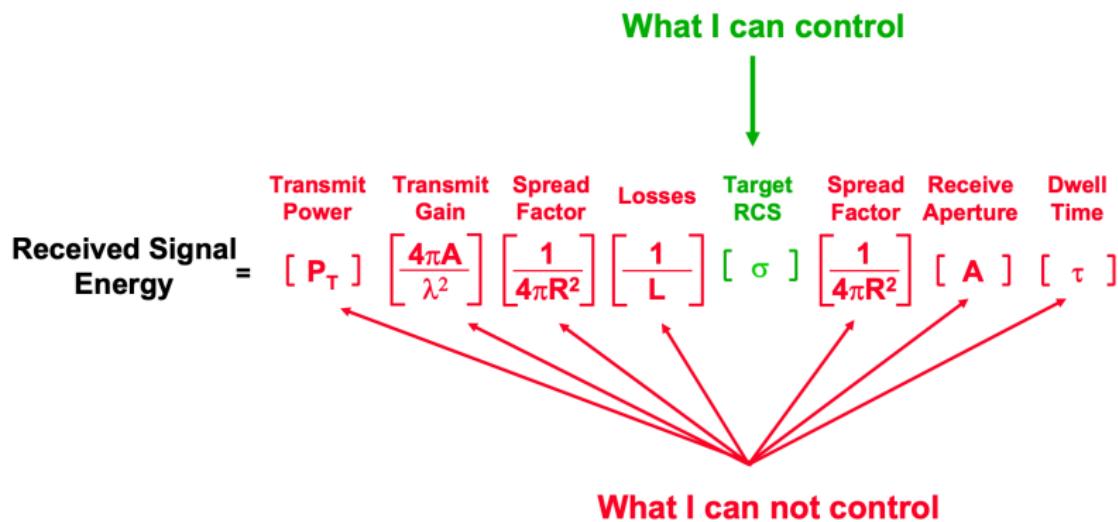
1 Radar Basics

2 CAD: Open Source

3 Mathematica

4 Backup Slides

Control Factors



Lab 2002



The Programmers Solid 3D CAD Modeller

The screenshot shows the OpenSCAD website. At the top left is a 3D model of a yellow sphere with a green ring attached. To its right is the text "OpenSCAD" in green and "The Programmers Solid 3D CAD Modeller" in black. On the far right is a button labeled "DONATE TO OUR COLLECTIVE" with a blue icon. Below the header is a navigation bar with links: home (which is highlighted), about, news, downloads, documentation, libraries, gallery, community, and github.



The Programmers Solid 3D CAD Modeller

The screenshot shows the official website for OpenSCAD. At the top, there's a navigation bar with links for home, about, news, downloads, documentation, libraries, gallery, community, and github. A prominent yellow button in the center says "Download OpenSCAD". Below the download button, there are four smaller boxes labeled Tutorial, Libraries, Books, and Cheat Sheet. To the left, there's a sidebar with sections for Recent News, Google Summer of Code 2024, OpenSCAD, Inc., Google Summer of Code 2022, and OpenSCAD on social media.



OpenSCAD is Pythonic

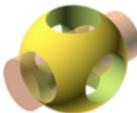
The screenshot shows the OpenSCAD 2015.03 application window. On the left is the 'Editor' pane displaying the SCAD source code for 'example009.scad'. On the right is the 'Viewport' showing a 3D perspective view of a mechanical part, specifically a fan assembly, with yellow highlights on certain parts. The bottom right corner of the viewport displays the version 'OpenSCAD 2015.03'.

```
1 bodywidth = dxf_dim(file = "example009.dxf", name = "bodywidth");
2 fanwidth = dxf_dim(file = "example009.dxf", name = "fanwidth");
3 platewidth = dxf_dim(file = "example009.dxf", name = "platewidth");
4 fan_side_center = dxf_cross(file = "example009.dxf", layer = "fan_side_center");
5 fanrot = dxf_dim(file = "example009.dxf", name = "fanrot");
6
7 // linear_extrude(height = bodywidth, center = true, convexity = 10)
8 // import(file = "example009.dxf", layer = "body");
9
10 % for (z = [-(bodywidth/2 + platewidth/2),
11 // -(bodywidth/2 + platewidth/2)]) {
12 // translate([0, 0, z])
13 // linear_extrude(height = platewidth, center = true, convexity = 10)
14 // import(file = "example009.dxf", layer = "plate");
15 }
16
17 intersection() {
18     linear_extrude(height = fanwidth, center = true, convexity = 10, twist = -fanrot)
19     import(file = "example009.dxf", layer = "fan_top");
20
21     // NB! We have to use the deprecated module here since the "fan_side"
22     // layer contains an open polyline, which is not yet supported
23     // by the import() module.
24     rotate_extrude(file = "example009.dxf", layer = "fan_side",
25                     origin = fan_side_center, convexity = 10);
26 }
```

Viewport: translate = [0.61 -1.31 -2.07], rotate = [55.00 0.00 25.00], distance = 142.23



OpenSCAD Tutorials



OpenSCAD

The Programmers Solid 3D CAD Modeller

[DONATE TO OUR COLLECTIVE](#)

home about news downloads documentation libraries gallery community github

Documentation

[OpenSCAD Tutorial](#)
[OpenSCAD User Manual](#)
[OpenSCAD Language Reference](#)
[Code Cheat Sheet](#)

Books

[English](#)
[German / Deutsch](#)
[Spanish / Español](#)

Videos

[OpenSCAD: Introduction](#)

Articles / Blogs

[How to use OpenSCAD](#)
[3D Spielplatz \(german\)](#)
[OpenSCAD Tutorial Series](#)

 Documentation  Books  Videos  Articles / Blogs

OpenSCAD Tutorial

Table of Contents

1. Chapter 1: A few words about OpenSCAD and getting started with the first object
2. Chapter 2: Scaling the model and first steps for parameterizing models
3. Chapter 3: Resizing models and more ways of combining objects
4. Chapter 4: Introducing modules to organize the code
5. Chapter 5: Using multiple scripts and libraries
6. Chapter 6: Control flow, conditional creation of objects
7. Chapter 7: Loops and creating more complex patterns
8. Chapter 8: Extruding 2D shapes into 3D objects
9. Chapter 9: Math, calculations and low level geometry creation

7 / 35

Daniel Topa

Design Tools for Satellite Models



OpenSCAD Libraries

 **OpenSCAD**
The Programmers Solid 3D CAD Modeler

[DONATE TO OUR COLLECTIVE](#)

[home](#) [about](#) [news](#) [downloads](#) [documentation](#) [libraries](#) [gallery](#) [community](#) [github](#)

Libraries

General

- [BOSL](#)
- [BOSL2](#)
- [desCAD](#)
- [NopCADlib](#)
- [UBS](#)

Functional OpenSCAD

Constructive

BOLTS

Asset Collection

Single Topic

- [Round Anything](#)
- [Mark's Enclosure Helper](#)
- [funcutils](#)
- [threads.scad](#)
- [Smooth Primitives Library](#)
- [Function Plotting Library](#)
- [ClosePoints Library](#)
- [Tray Library](#)
- [YAPP Generator](#)
- [STEMHIE Parts Library](#)
- [Catch'n'Hole](#)
- [Pathbuilder](#)
- [Altair's 2D Library](#)

BOSL

The Befify OpenScad Library - A library of tools, shapes, and helpers to make OpenScad easier to use.

- » [Library](#)
- » [Documentation](#)
- » [License: BSD-3-Clause](#)

BOSL2 (beta)

Befify OpenScad Library v2 - A library of tools, shapes, and helpers to make OpenScad easier to use.

- » [Library](#)
- » [Documentation](#)
- » [Tutorials](#)
- » [License: BSD-3-Clause](#)

desCAD

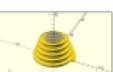
Reduce the burden of 3D modeling in mathematics.

- » [Library](#)
- » [Documentation](#)
- » [License: LGPL-3.0-only](#)

NopCADlib

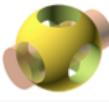
An ever expanding library of parts modelled in OpenSCAD useful for 3D printers and enclosures for electronics, etc.

- » [Library](#)
- » [Documentation](#)
- » [License: GPL-3.0-or-later](#)





OpenSCAD Book



OpenSCAD

The Programmers Solid 3D CAD Modeler

[DONATE TO OUR COLLECTIVE](#)

home about news downloads documentation libraries gallery community github

Documentation

- OpenSCAD Tutorial
- OpenSCAD User Manual
- OpenSCAD Language Reference
- Code Cheat Sheet

Books

- English
- French / Français
- German / Deutsch
- Spanish / Español

Videos

- OpenSCAD: Introduction

Articles / Blogs

- How to use OpenSCAD
- 3D Spielplatz (german)
- OpenSCAD Tutorial Series
- Tutorial at EduTechWiki

 Documentation

 Books

 Videos

 Articles / Blogs

English

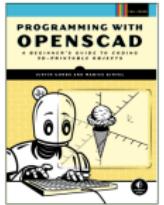
Programming with OpenSCAD - A Beginner's Guide to Coding 3D-Printable Objects

This book channels OpenSCAD's visual benefits and user-friendliness into a STEAM-focused, project-based tutorial that teaches the basics of coding, 3D printing, and computational thinking while you develop your spatial reasoning by creating 3D designs with OpenSCAD.

Accessibly written for a wide audience (advanced middle schoolers, high school students, college students, artists, makers and lifelong-learners alike), this is the perfect guide to becoming proficient at programming in general and 3D modeling in particular.

Author: Justin Gohde & Marius Kintel
Publisher: No Starch Press
Date: July 2021

Read more at programmingwithopenscad.github.io





OpenSCAD Cheat Sheet

<p>Syntax</p> <pre>var = value; var = cond 2.value_if_true + value_if_false; var = function (x) { x * x; } node.name(.) = []</pre> <p>name();</p> <pre>function name(.) = - name(); include <...scad>; use <...scad>;</pre>	<p>Modifier Characters</p> <ul style="list-style-type: none"> - disable ! show only # highlight / debug % transparent / background 	<p>Lists</p> <pre>list = [value,...]; create a list var = list[2]; index a list (from 0) var = list.; dot notation indexing (x/y/z)</pre>	<p>Functions</p> <pre>concat lookup str chr ord search version version_num parent_id(id)</pre>
<p>Constants</p> <pre>PI = undefined value PI = mathematical constant ≈ (-3.14159)</pre>	<p>2D</p> <pre>circle((radius d=diameter) size([size,center]) source([width,height],center) solving([points]) solving([points,paths]) text([text, size, font, tip, align, spacing, direction, language, script]) import([<i>file</i>, convert]) acollection(cut))</pre>	<p>Boolean operations</p> <pre>union() difference() intersection()</pre>	
<p>Operators</p> <pre>n + n Addition n - n Subtraction n * n Multiplication n / n Division n % n Modulo n ** n Exponentiation n < n Less Than n ≤ n Less or Equal b == b Equal b != b Not Equal n ≥ n Greater or Equal n > n Greater Than b & b Logical And b b Logical Or !b Negation</pre>	<p>3D</p> <pre>sphere((radius d=diameter) center([size,center]) cap([width,depth,height],center) cylinder([height,radius,center]) cylindre([h,radius,center]) extrude([points, faces, convexity]) inward([-ext, convexity]) linear_extrude([height,center,convexity,twist,slices]) rotate_extrude([angle,convexity]) surface([file = "...",ext,center,convexity])</pre>	<p>List Comprehensions</p> <pre>generate [for (i = range([list]) i] generate [for (i; condition(i); next i] flatten [each i] conditions [for (i = ...) if (condition(i)) i] conditions [for (i = ...) if (condition(i)) else y] assignments [for (i = ...) let (assignments) = i]</pre>	<p>Mathematical</p> <pre>abs sign size cos sin tan acos asin atan atan2 fmod round ceil lo len lat log one sort err cada nil max norm cross</pre>
<p>Special variables</p> <pre>\$fa minimum angle \$fs minimum size \$fn number of fragments \$st animation step \$View viewport rotation angles in degrees \$View viewport translation \$View viewport camera distance \$View viewport camera field of view \$Children number of module children \$Preview true in FS preview, false for F6</pre>	<p>Transformations</p> <pre>translate([x,y,z]) rotate([x,y,z]) rotate_a([x,y,z]) scale([x,y,z]) resin([x,y,z],auto,convexity) mirror([x,y,z]) matrix3d(n) matrix4d(n) color("colorname",alpha) color("hexvalue") color([r,g,b,a]) offset([delta,chanfer]) hull() minkowski(convexity)</pre>	<p>Flow Control</p> <pre>for (i = [startend]) { ... } for (i = [startend]; condition() { ... } for (i = [...]-->) { ... } for (i = ...; j = ...; i = ...) { ... } intersectionFor[i = [startend]] { ... } intersectionFor[i = [startstepend]] { ... } intersectionFor[i = [...]--] { ... } if (...) { ... } let (...) { ... }</pre>	
	<p>Type test functions</p> <pre>is.undefined is.boolean is.number is.string is.list is.function</pre>	<p>Other</p> <pre>sgdl(-) render(convexity) children([id]) assert(condition, message) assess(-){-[-]}</pre>	



Sample Texts

The screenshot shows the Blender 4.2 LTS website. At the top, there's a navigation bar with links for Features, Download, Support, Get Involved, About, Jobs, and a blue 'Donate' button. Below the navigation is a large, vibrant image of a multi-colored fish swimming in an underwater environment. Overlaid on this image is the text 'Blender 4.2 LTS' in a large, bold, white font. Below this, a smaller text reads 'Packed with features and improvements ready to power your projects for the next two years.' At the bottom left, there are two buttons: 'Download' and 'What's New'. A small 'by Blender Studio' text is at the bottom right.

User Manual

Developer Documentation

Python API



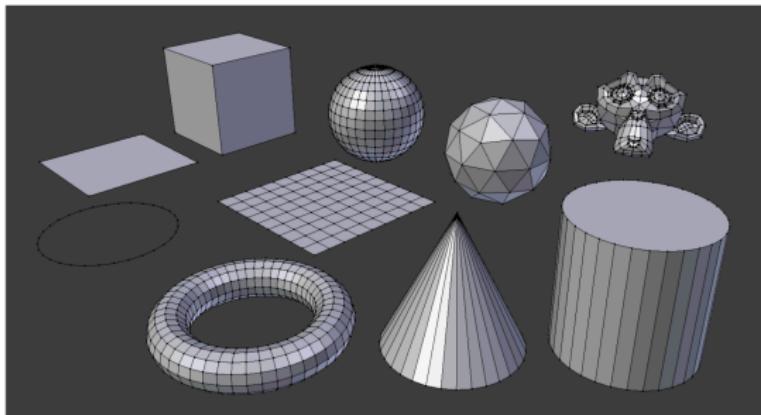


Primitives

Primitives



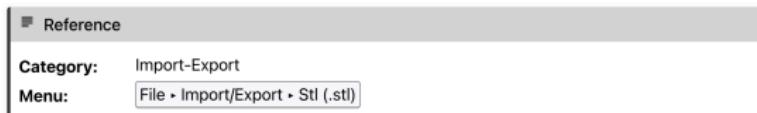
A common object type used in a 3D scene is a mesh. Blender comes with a number of "primitive" mesh shapes that you can start modeling from. You can also add primitives in Edit Mode at the 3D cursor.





Stereolithography File Format

STL



The STL-file format is useful if you intend to import/export the files for CAD software. It is also commonly used for loading into 3D printing software.

Importing

General

Scale

Value by which to scale the imported objects in relation to the world's origin.

Scene Unit

Apply current scene's unit (as defined by unit scale) to imported data.

Forward / Up Axis

Since many applications use a different axis for pointing upwards, these are axis conversion for these settings. Forward and up axes – By mapping these to different axes you can convert rotations between applications default up and forward axes.

Blender uses Y forward, Z up (since the front view looks along the +Y direction). For example, it is common for applications to use Y as the up axis, in that case -Z forward, Y up is needed.



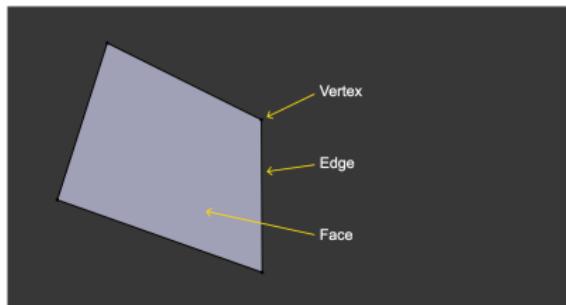


Mesh Basics



Structure

With meshes, everything is built from three basic structures: *vertices*, *edges* and *faces*.



Example of mesh structure. #



Normals

Editing Custom Split Normals

≡ Reference

Mode:	Edit Mode
Menu:	Mesh • Normals
Shortcut:	Alt-N

There are a number of tools for editing custom split normals. The custom normal mesh edit tools can affect all normals (the default), or only selected ones. To select a custom normal associated with a particular vertex and face:

- Make the element selection mode both Vertex and Face (use `Shift-LMB` to enable the second one).
- Select one or more vertices, then select a face. This can be repeated to select more vertices and a different face and so on. It is easiest to see the effect of these tools if you turn on the Edit Mode Overlays option *Display vertex-per-face normals as lines*.

See also

[Editing Normals.](#)

Importing Custom Split Normals

Some tools, particularly those used in CAD, tend to generate irregular geometry when tessellating their objects into meshes (very thin and long triangles, etc.). Auto-computed normals on such geometry often gives bad artifacts, so it is important to be able to import and use the normals as generated by the CAD tool itself.

Note

Currently, only the [FBX Importer](#) and [Alembic Importer](#) are capable of importing custom normals.



Wolfram - née Mathematica


WOLFRAM COMPUTATION MEETS KNOWLEDGE

Products & Services ▾ Technologies ▾ Solutions ▾ Learning & Support ▾ Company ▾ Q Search WolframAlpha.com | WolframCloud.com | All Sites & Public Resources.

Computation. Data. Decisions. JUST USE WOLFRAM

Technology Consulting Education Wolfram AI



QUANTUM COMPUTING
FINANCIAL RISK MANAGEMENT
IMAGE ANALYSIS
ARTIFICIAL INTELLIGENCE
MORE...
CHEMICAL ENGINEERING
CONTROL SYSTEMS

 Just Use Wolfram  Wolfram Language  Wolfram|Alpha Intelligence  Our Experts, Your Projects  Transforming Education  Leading Innovation in Science & Tech



C-3PO of Information Interchange

GUIDE

Listing of All Formats

The Wolfram Language supports many formats, with many subformats, variants, and options.

"3DS" — 3D Studio format (.3ds)

"7z" — 7z compression and archival format (.7z)

"ACO" — Adobe color palette format (.aco)

"Affymetrix" — Affymetrix microarray formats (.cel, .cdf, .chp, .gin, .psi)

"AgilentMicroarray" — Agilent microarray data format (.txt)

"AIFF" — AIFF Macintosh sound format (.aif, .aiff)

"ApacheLog" — Apache access log file format

"ArcGRID" — ESRI GIS format

"AU" — μ law encoding Unix Audio Format (.au)

"AVI" — Microsoft AVI format (.avi)



"Base64" — Base64 ASCII encoding

"BDF" — BDF physiological signal recordings format (.bdf)

"Binary" — sequence of binary data types

"Bit" — sequence of binary bits

"BMP" — Microsoft bitmap format (.bmp)

"BSON" — JSON-like binary format (.bson)

"Byte" — sequence of 8-bit unsigned integers

"BYU" — BYU 3D geometry format (.byu)

"BZIP2" — bzip2 compression format (.bz2)

"C" — C code generation (.c)

"CDED" — Canadian digital elevation data (.dem)

"CDF" — Wolfram Computable Document Format (.cdf)

"CDX" — ChemDraw Exchange format (.cdx)

"CDXML" — ChemDraw Exchange XML format (.cdxml)

"Character8" — sequence of 8-bit characters

"Character16" — sequence of 16-bit Unicode characters

"CIF" — CIF molecular model format (.cif)

"CML" — Chemical Markup Language (.cml)

"Complex64" — IEEE single-precision complex numbers

"Complex128" — IEEE double-precision complex numbers

"Complex256" — IEEE quad-precision complex numbers

"CSV" — comma-separated values (.csv)

"Cube" — Gaussian Cube file (.cube)

"CUR" — Windows cursor resource format (.cur)

"DAE" — COLLADA digital asset exchange format (.dae)

"DBF" — dBase database file (.dbf)

"DICOM" — DICOM medical image format (.dcm, .dic)

"DICOMDIR" — directory of DICOM files (DICOMDIR)

"DIF" — VisiCalc data interchange format (.dif)

"DIMACS" — DIMACS graph data format (.col, .col.b)

"Directory" — filesystem directory hierarchy

"DOCX" — Microsoft Word format import

"DOT" — DOT graph data format (.gv, .dot)

"DTA" — Stata database transport format (.dta)

"DXF" — AutoCAD 2D and 3D format (.dxf)

"EDF" — EDF physiological signal recordings format (.edf)

"EML" — mail message format

"EPS" — Encapsulated PostScript (.eps)

"Expression.JSON" — JSON for representing expressions (.json)

"ExpressionML" — Wolfram Language expression XML format (.xml)

"FASTA" — bioinformatics sequence format (.fasta, .fa, .fsa, .mpfa)

"FASTQ" — FASTQ molecular biology format (.fastq, .fq)

"FCHK" — Formatted Checkpoint file (.fchk)

"FCS" — FCS molecular biology format (.fcs, .lmd)

"FITS" — FITS astronomical data and image format (.fit, .fits)

"FLAC" — FLAC lossless audio codec (.flac)

"FLV" — Adobe/Macromedia Flash video format (.flv)



C-3PO of Information Interchange II

"GaussianLog" — Gaussian log file (.log)

"GenBank" — GenBank molecular biology format (.gb, .gbk)

"GeoJSON" — geographic data format based on JSON (.geojson)

"GeoTIFF" — annotated TIFF raster files (.tif)

"GIF" — static or animated GIF (.gif)

"GPX" — GPX global positioning data format (.gpx)

"Graph6" — dense graph format (.g6)

"Graphlet" — GML graph data (.gml)

"GraphML" — GraphML graph format (.graphml)

"GRIB" — GRIB scientific data format (.grb, .grb)

"GTOP030" — USGS global topographic data (.dem)

"GXL" — GXL graph data (.gxl)

"GZIP" — Unix GZIP compression (.gz)

"HarwellBoeing" — Harwell-Boeing sparse matrix format

"HDF" — NCSA hierarchical data format (.hdf)

"HDF5" — NCSA HDF 5 hierarchical data format (.h5)

"HEIF" — High Efficiency Image Format (.heic, .heif)

"HIN" — HyperChem HIN data format (.hin)

"HTML" — HTML format (.htm, .html), including embedded GIFs, CSS, etc.

"HTMLFragment" — HTML fragment (.htm, .html)

"HTTPRequest" — HTTP request as transmitted to a web server

"HTTPResponse" — HTTP response as generated by a web server

"ICC" — ICC color profile format (.icc, .icm)

"ICO" — Windows icon resource format (.ico)

"ICS" — iCalendar format (.ics, .ical, .ifb)

"ICNS" — Macintosh icons format (.icns)

"INI" — key-value configuration format (.ini)

"Integer8" — sequence of 8-bit signed integers

"Integer16" — sequence of 16-bit signed integers

"Integer24" — sequence of 24-bit signed integers

"Integer32" — sequence of 32-bit signed integers

"Integer64" — sequence of 64-bit signed integers

"Integer128" — sequence of 128-bit signed integers

"ISO" — ISO archival format (.iso)

"JavaProperties" — Java key-value format (.properties)

"JavaScriptExpression" — JSON with objects as associations (.json)

"JAMP-DX" — chemical spectroscopy format (.jdx, .dx, .jcm)

"JPEG" — JPEG raster image format (.jpeg, .jpg)

"JPEG2000" — JPEG 2000/JP2 raster image format (.jp2, .j2k)

"JWK" — JavaView format (.jwk)

"JSON" — JSON web service description (.json)

"KML" — Google Earth GIS format (.kml, .kmz)

"LaTeX" — LaTeX format (.latex)

"LEDA" — LEDA graph data format (.gw, .lgr)

"List" — numbers or strings, one per line

© 2018 Daniel Topa | Last updated 2018-06-01 | Version 1.1 (v1.1)

"MAT" — MATLAB/Octave data format (.mat)

"MathML" — MathML math content format

"Matroska" — Matroska multimedia container format (.mkv)

"Maya" — Maya entity files (.ma)

"MBOX" — MBOX Unix mailbox format (.mbox)

"MCTT" — Modelica "CombiTimeTable" data format (.txt)

"MDB" — Microsoft Access database file (.mdb)

"MGF" — Wolfram Mathematica MGF bitmap format (.mgf)

"MIDI" — MIDI music format (.mid)

"MMCIF" — MMCIF 3D molecular model format (.cif)

"MO" — Modelica model format (.mo)

"MOBI" — Mobipocket ebook format import

"MOL" — MDL MOL molecular models and drawings format (.mol)

"MOL2" — Tripos MOL2 format (.mol2)

"MP3" — MP3 audio format (.mp3)

"MP4" — MPEG-4 Part 14 video format (.mp4)

"MPS" — MPS linear programming system format (.mps)

"MTP" — Minitab portable worksheet format (.mtp)

"MTX" — Matrix Market sparse matrix format (.mtx)

"MX" — Wolfram Language serialized package format (.mx)

"MXNet" — MXNet net representation format (.json, .params)



C-3PO of Information Interchange III

"NASACDF" — NASA common data format (.cdf)

"NB" — Wolfram Notebook format (.nb)

"NDK" — NDK seismographic data format (.ndk)

"NetCDF" — Unidata scientific data format

"NEXUS" — NEXUS phylogenetic format (.nex, .nxs)

"NOFF" — 3D object file format with normals (.noff, .cnoff)

"OBJ" — Wavefront OBJ format (.obj)

"ODS" — OpenDocument spreadsheet format (.ods)

"OFF" — 3D object file format (.off, .coff)

"Ogg" — Ogg multimedia container format (.ogg)

"OpenEXR" — OpenEXR raster image format (.exr)

"Pajek" — Pajek graph data format (.net)

"PBM" — ASCII portable bitmap format (.pbm)

"PCAP" — Network packet capture format (.pcap)

"PCX" — Windows Paintbrush format (.pcx)

"PDB" — Protein Data Bank format (.pdb)

"PDF" — Adobe Acrobat PDF format (.pdf)

"PGM" — ASCII portable graymap format (.pgm)

"PHPIni" — PHP-enabled configuration format (.ini)

"PLY" — PLY 3D geometry format (.ply)

"PNG" — PNG raster image format (.png)

"PNM" — ASCII portable raster format (.pnm)

"POV" — POV-Ray ray-tracing object description format (.pov)

"PPM" — ASCII portable color pixmap format (.ppm)

"QuickTime" — Apple QuickTime multimedia container format (.mov)

"RAR" — RAR compression and archival format (.rar)

"Raw" — raw image format

"RawBitmap" — pure binary bitmap data

"Raw.JSON" — JSON with objects as associations (.json)

"RData" — R data format family import

"RDS" — R data format family import

"Real32" — IEEE single-precision real numbers

"Real64" — IEEE double-precision real numbers

"Real128" — IEEE quad-precision real numbers

"RIB" — Renderman interchange format (.rib)

"RLE" — run-length encoding (.rle)

"RSS" — RSS feed format (.rss)

"RTF" — Microsoft Rich Text Format (.rtf)

"SAS7BDAT" — SAS database transport format (.sas7bdat)

"SAV" — SPSS database transport format (.sav, .zsav)

"SCT" — Scitex CT prepress format (.sct)

"SDF" — MDL SDF format (.sdf)

"SDTS" — SDTS DLG and DEM spatial data files

"SDTSDEM" — digital elevation files

"SFF" — SFF molecular biology format (.sff)

"SHP" — ESRI shape file format (.shp)

"SMA" — SystemModeler model archive format (.sma)

"SME" — SystemModeler experiment format (.sme)

"SMILES" — SMILES chemical format (.smi)

" SND" — SND file format (.snd)

"SP3" — geospatial data format (.sp3)

"Sparse6" — sparse graph format (.s6)

"STL" — stereolithography format (.stl)

"String" — arbitrary data as a Wolfram Language string

"SVG" — Scalable Vector Graphics format (.svg)

"SurferGrid" — geospatial data format (.grd)

"SXC" — OpenOffice 1.0 spreadsheet file (.sxc)



C-3PO of Information Interchange IV

"Table" — arbitrary tabular data (.dat)

"TAR" — Unix TAR archive (.tar)

"TerminatedString" — null-terminated string

"TeX" — TeX document format (.tex)

"TeXFragment" — TeX fragment

"Text" — plain text (.txt)

"TGA" — TrueVision Targa format (.tga)

"TIGER" — TIGER/Line GIS format

"TLE" — TLE satellite data format (.tle, .tce)

"TGF" — TGF graph format (.tgf)

"TIFF" — TIFF raster image format (.tiff, .tif)

"TSV" — tab-separated values (.tsv)

"UBJSON" — Universal Binary JSON format (.ubj)

"UnsignedInteger8" — sequence of 8-bit unsigned integers

"UnsignedInteger16" — sequence of 16-bit unsigned integers

"UnsignedInteger24" — sequence of 24-bit unsigned integers

"UnsignedInteger32" — sequence of 32-bit unsigned integers

"UnsignedInteger64" — sequence of 64-bit unsigned integers

"UnsignedInteger128" — sequence of 128-bit unsigned integers

"USGSDEM" — USGS ASCII DEM digital elevation file (.dem)

"UUE" — Unix uuencode format (.uue, .enc)

"VCF" — vCard format (.vcf)

"VCS" — vCalendar format (.vcs)

"VideoFrames" — sequence of raster images

"VRML" — Virtual Reality Modeling Language format (.vrml)

"VTK" — Visualization Toolkit 3D format (.vtk)

"WARC" — Web ARchive format (.warc)

"WAV" — Microsoft WAV audio format (.wav)

"Wave64" — Sony Wave64 audio format (.w64)

"WDX" — Wolfram data exchange format (.wdx)

"WebP" — WebP raster image format (.webp)

"WL" — Wolfram Language package source (.wl)

"WMLF" — Wolfram machine learning format (.wmlf)

"WLNet" — Wolfram Language neural net format (.wlnet)

"WXF" — Wolfram exchange format (.wxif)

"X3D" — X3D XML geometry format (.x3d)

"XBM" — X bitmap format (.xbm)

" XHTML" — XML-syntax HTML (.xhtml)

"XHTMLMathML" — XHTML with embedded MathML

"XLS" — Excel spreadsheet format (.xls)

"XLSX" — Microsoft Excel 2007 format (.xlsx)

"XML" — arbitrary XML (.xml)

"XPORT" — SAS interchange format (.sas, .xpt)

"XYZ" — XYZ molecule geometry file (.xyz)

"ZIP" — Windows ZIP archive (.zip)

"ZPR" — Z Corp 3D printer format (.zpr)

"ZSTD" — Zstandard format (.zstd)



3D Geometry and Modeling Formats

WOLFRAM Products & Services Technologies Solutions Learning & Support Company Search

Wolfram Language & System Documentation Center Search Wolfram Language Home Page Functions Related Guides Tech Notes

3D Geometry & Modeling Formats

The Wolfram Language supports import and export of 3D geometry from all standard formats—with its symbolic representation of 3D objects allowing immediate faithful interchange.

3D Object Geometry Formats

- "**PLY**" — PLF 3D geometry format (.ply)
- "**DAE**" — COLLADA digital asset exchange format (.dae)
- "**OFF**", "**OFF++**" — 3D object file formats (.off, .off+, .onoff)
- "**BYU**" — BYU 3D geometry format (.byu)
- "**OBJ**" — Wavefront OBJ format (.obj)
- "**VTK**" — Visualization Toolkit 3D format (.vtk)

3D Viewing Formats

- "**X3D**" — X3D XML geometry format (.x3d)
- "**JWZ**" — JavaView format (.jwz)
- "**VRML**" — Virtual Reality Modeling Language format (.vrml)

Modeling & Rendering Formats

- "**Maya**" — Maya entity files (.ma)
- "**POV**" — POV-Ray ray-tracing object description format (.pov)
- "**LW**" — LightWave 3D file format (.lws)
- "**3DS**" — 3D Studio format (.3ds)
- "**RIB**" — Renderman interchange format (.rib)

CAD-Related Formats

- "**DXF**" — AutoCAD 2D & 3D formats (.dwf)
- "**STL**" — stereolithography format (.stl)
- "**ZPR**" — Z Corp. 3D printer format (.zpr)



Algorithmically Generate 3D Models

↳ 3D Printing

Algorithmically Generate 3D Models

The Wolfram Language core principles of automation and unification make it easy to generate geometry and 3D-printable models algorithmically better than ever.

Create 3D-printable objects from 2D images.

```
In[1]:= extrudeImage[image_]:=  
  Block[{res, img},  
    img = DeleteSmallComponents[Binarize[image, 0.9], 500];  
    res = ImageMesh[ColorNegate[img]];  
    RegionProduct[res, Line[{{0, 1}, {50, 1}}]]  
  ]
```

```
In[2]:= extrudeImage /@ {, , , , 
```

```
Out[2]= {, , , , 
```



Mesh Repair

3D Printing

Automatically Detect Defects and Repair Meshes

Version 11 integrates fully automated detection of mesh defects in 3D models and provides repair functionality.

HoleEdges edges around a hole in the surface	TJunctionEdges edges that form a T-junction
TinyFaces faces with near-zero area	OverlappingFaces faces that overlap
IsolatedVertices vertices without incident edges	DanglingEdges edges without incident faces
SingularEdges edges with more than two incident faces	SingularVertices vertices with a non-disc neighborhood
TinyComponents tiny connected mesh components	FlippedFaces faces that point inward

Find defects in a 3D model.

```
In[1]:= mesh = ExampleData[{"Geometry3D", "StanfordBunny"}, "Region"];
FindMeshDefects[mesh]
```

Out[1]:=



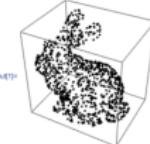
Reconstruct Models from 3D Data

3D Printing

Reconstruct Models from 3D Data

Version 11 includes state-of-the-art surface reconstruction from arbitrary 3D data.

```
In[1]:= pointcloud = ExampleData[{"Geometry3D", "StanfordBunny"}, "VertexData"];
Graphics3D[Point[RandomSample[pointcloud, 1000]]]
```

Out[1]= 

Reconstruct a 3D-printable Stanford bunny.

```
In[2]:= ListSurfacePlot3D[pointcloud, MaxPlotPoints -> 50, Axes -> None,
Boxed -> False, Mesh -> None]
```

Out[2]= 



Properties of Models, Materials, and Costs

↳ 3D Printing

Get Properties of Models, Materials, and Costs

Leveraging built-in geometric computation and the extensive Wolfram Knowledgebase, properties of 3D models and materials can be computed.

```
In[1]:= model = ExampleData[{"Geometry3D", "SpaceShuttle"}, "Region"]
```

Out[1]:



Mass of the model in aluminum.

```
In[2]:= volume = Quantity[Volume[model], "Centimeters"^3]
```

Out[2]: 55.5217 cm³

```
In[3]:= aluminum (element) \[Equal] Density\[Function]volume
```

Out[3]: 149.909 g



Import 3D Models

3D Printing

Import 3D Models

Version 11 supports import and export of 3D geometry from all standard formats.

STL	PLY	OBJ	OFF	DXF	DAE	ZPR	VTK	BYU
X3D	JVX	VRML	Maya	POV	LWO	3DS	RIB	XYZ

Import the Bust of Eleonora Duse—at the Gallery of Modern Art of the Palazzo Pitti in Florence, Italy—from My Mini Factory repository.

```
Import["https://myminifactory.net/uploads/objects/files/1407dfe1e1a0cb2fe013fe1ffff3fea31cbd224.stl"]
```

Out[1]=



Ready-Made 3D-Printable Models

3D Printing

Ready-Made 3D-Printable Models

Direct access to a large volume of curated 3D-printable objects, specially organized and created for the Wolfram Language.

In[1]:= `skull [anatomical structure]` `["Region"]`

Out[1]:=

From geographical features to knots.

show complete Wolfram Language input



Animation with Blender

The screenshot shows a post on the Wolfram Community. The header includes the Wolfram logo, the text "WOLFRAM COMMUNITY", and "Connect with users of Wolfram technologies to learn, solve problems and share ideas". There are "Join" and "Sign In" buttons. Below the header is a navigation bar with "Dashboard", "Groups" (which is selected), "People", and a search bar. Underneath the navigation is a social sharing section with a thumbs up icon (5 likes), "5213 Views", "1 Reply", "5 Total Likes", "View groups...", "Follow this post", and a "Share" button.

Animating Wolfram surfaces with Blender: Mesh deforming & Superellipse



Guenther Gsaller, Retired

Posted 2 years ago



Animating Wolfram surfaces with Blender: Mesh deforming & Superellipse
by Guenther Gsaller

This is a continuation to the posts listed at: <https://community.wolfram.com/web/ggoff/>





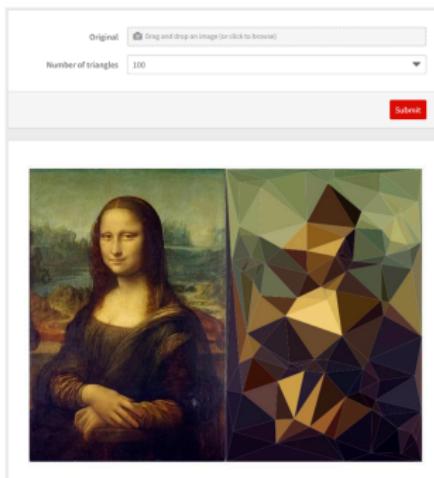
Triangulation of paintings with Delaunay mesh

Automating triangulation of paintings with Delaunay
mesh and cloud app



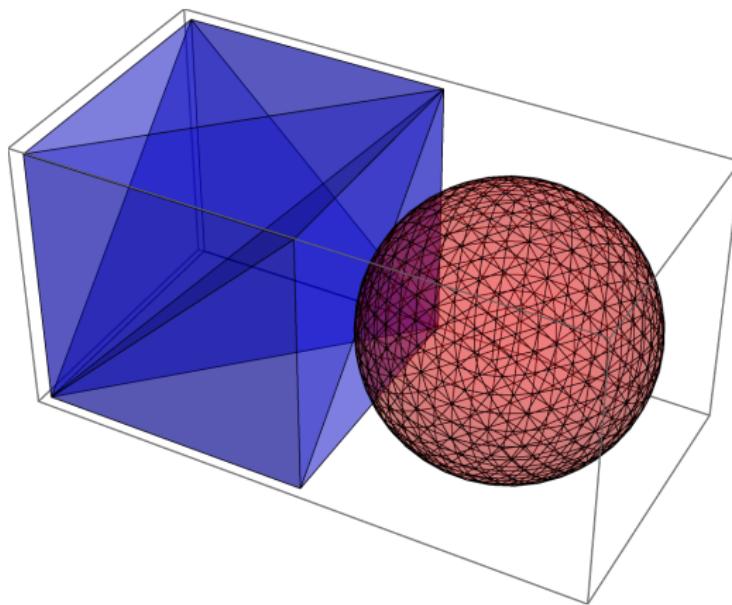
Kirill Belov

Posted 2 years ago





Simple Shapes

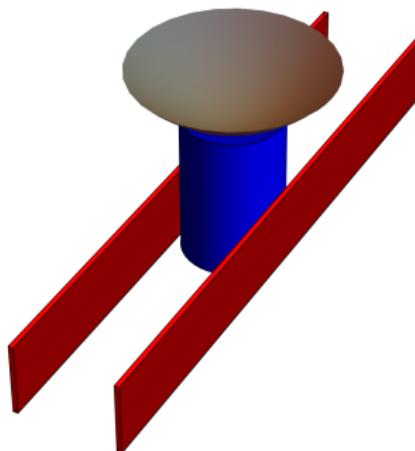




Simple Shapes



DanielSat C





Professional Societies: Computational Mechanics





Bibliography I

- [1] MIT Lincoln Lab. “Target Radar Cross Section”. In: Introduction to Radar Systems. MIT Lincoln Lab. MIT Lincoln Lab, 2002, p. 45. URL:
<https://www.ll.mit.edu/sites/default/files/outreach/doc/2018-07/lecture%204.pdf>.



Design Tools for Satellite Models

Daniel Topa
daniel.topa@hii-tsd.com

Huntington Ingalls Industries
Mission Technologies

November 14, 2024