Outils logiciels pour la 3D

(page créée le 28 fév 2022)

En ligne

imagetoSTL https://imagetostl.com/
Conversion d'images en niveau de gris / noir et blanc en fichier STL.

sculptGL https://stephaneginier.com/sculptgl/
Modelage à la souris de modèles 3D (exporte en STL)

3DSlash https://www.3dslash.net/
Modelage sur un bloc «à la minecraft» (l'export en STL nécessite un compte)

thingiverse https://www.thingiverse.com/
Plateforme d'objets 3D sous licence libre

BlocksCAD https://www.blockscad3d.com/
Langage à bloc (type scratch, blockly) pour la modélisation 3D

TinkerCAD https://www.tinkercad.com/

Logiciels

Blender

https://www.blender.org/

openSCAD

Voir OpenSCAD

Wings3D

Voir Wings3D

FreeCAD

https://www.freecadweb.org/

GMSH

Visualiser des objets STL et bien plus (édition de mesh). http://gmsh.info/

Pour afficher les surfaces : Tools → Options → Mesh → Visibility → Surface faces

MeshLab

https://www.meshlab.net/

Scripting

Dimensions d'un objet STL

Ce script en python3 donne les dimensions de la boîte englobante de l'objet :

stldim.py (cliquer pour afficher le code)

stldim.py

```
#!/usr/bin/python

# Script python pour trouver les dimensions de la boîte englobante d'un objet STL
# nb : les fichiers STL n'ont pas d'unité réelle, mais en général les logiciels qui ont
#      besoin d'une unité, attribuent 1mm pour 1 en unité STL
# source : https://www.reddit.com/r/3Dprinting/comments/7ehlfc/python_script_to_find_stl_dimensions/
#
# Testé avec Python 3.5.3 + numpy-stl 2.16.3 @ kirin / Debian 9.5
#
# Requirements: sudo pip install numpy-stl

import math
import stl
from stl import mesh
import numpy

import os
import sys

if len(sys.argv) < 2:
    sys.exit('Usage: %s [stl file]' % sys.argv[0])

if not os.path.exists(sys.argv[1]):
    sys.exit('ERROR: file %s was not found!' % sys.argv[1])

# this stolen from numpy-stl documentation
# https://pypi.python.org/pypi/numpy-stl

# find the max dimensions, so we can know the bounding box, getting the height,
# width, length (because these are the step size)...
def find_mins_maxs(obj):
    minx = maxx = miny = maxy = minz = maxz = None
    for p in obj.points:
        # p contains (x, y, z)
        if minx is None:
            minx = p[stl.Dimension.X]
            maxx = p[stl.Dimension.X]
            miny = p[stl.Dimension.Y]
            maxy = p[stl.Dimension.Y]
            minz = p[stl.Dimension.Z]
            maxz = p[stl.Dimension.Z]
        else:
            maxx = max(p[stl.Dimension.X], maxx)
            minx = min(p[stl.Dimension.X], minx)
            maxy = max(p[stl.Dimension.Y], maxy)
            miny = min(p[stl.Dimension.Y], miny)
            maxz = max(p[stl.Dimension.Z], maxz)
            minz = min(p[stl.Dimension.Z], minz)
    return minx, maxx, miny, maxy, minz, maxz

main_body = mesh.Mesh.from_file(sys.argv[1])

minx, maxx, miny, maxy, minz, maxz = find_mins_maxs(main_body)

# the logic is easy from there

print ("File:", sys.argv[1])
print ("X:", maxx - minx)
print ("Y:", maxy - miny)
print ("Z:", maxz - minz)
```