Módulo 5: Software

Lady Johanna Trejos Hernández

Andrés Felipe Betancurt Rivera

Docente Ramiro Andrés Barrios Valencia

Universidad Tecnológica de Pereira

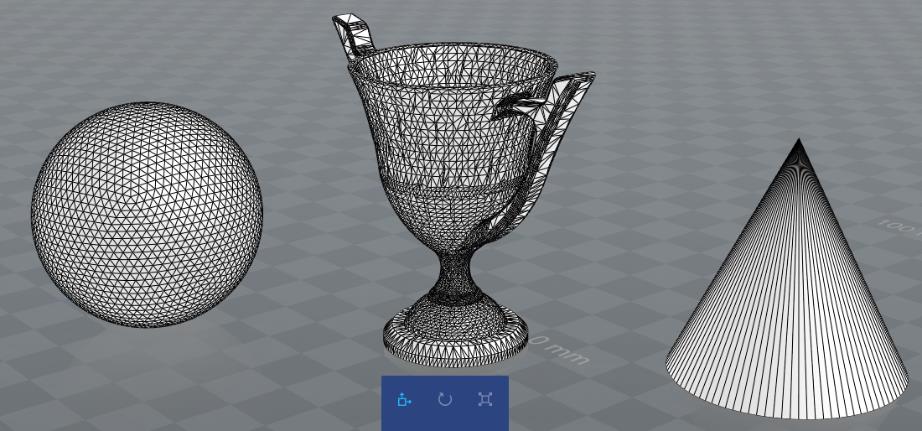
Facultad de Ingenierías FEECC

Asignatura Laboratorio de Electrónica Digital

14 de mayo de 2017

Para este módulo se desarrolló un aplicativo en el lenguaje Python para la lectura de archivos en formato STL y permitir el envío de las coordenadas al dispositivo controlador.

STL (*Standard Triangle Language*) es un formato de archivo informático de diseño asistido por computadora (*CAD*) que define geometría de objetos 3D, excluyendo información como color, texturas o propiedades físicas. Utiliza una malla de triángulos cerrada para definir la forma de un objeto, como se muestra en la siguiente imagen:



Un archivo STL puede ser exportado en formato ASCII o binario, siendo este último el más común por ser más compacto.

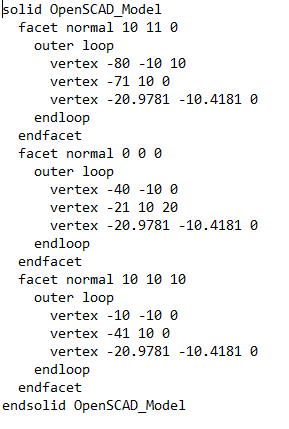
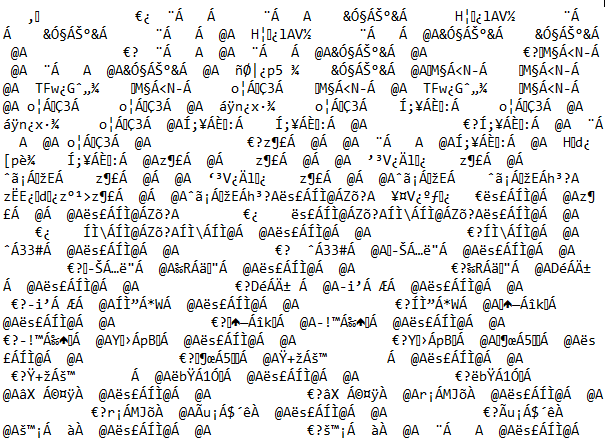
 

Ilustración . Archivo STL ASCII Ilustración . Archivo STL binario

Debido a que se está desarrollando es una impresora 2D de eje x-y, se cuenta con la necesidad de convertir la figura 3D entregada en el archivo STL a una figura en el plano x-y.

Para esto en el aplicativo, con ayuda de la librería numpy-stl de Python se realizó la lectura del archivo y posteriormente la búsqueda de los triángulos cuyos vértices estuvieran sobre el plano z <= 0. El código para este procedimiento se indica a continuación:

import serial

import numpy as np

import stl, math

from stl import mesh

import matplotlib.pyplot as plt

from scipy.spatial import ConvexHull

obj = mesh.Mesh.from\_file('archivo.stl')

points = set()

for p in obj.points:

if (p[2] and p[5] and p[8]) <= 0:

points.add((p[0],p[1]))

points.add((p[3],p[4]))

points.add((p[6],p[7]))

points = [list(p) for p in points]

points.sort()

Se ordenan los puntos de menor a mayor y a cada uno de ellos se le halla el vecino más cercano y así guardarlos en el orden correcto para ser enviados al controlador.

def distance(a,b):

if a == b:

return 1000

return math.sqrt( (a[0] - b[0])\*\*2 + (a[1] - b[1])\*\*2)

Distancias = [[distance(p, q) for q in points] for p in points]

my\_points = []

my\_points.append(points[0])

El último paso es enviar las coordenadas obtenidas de forma serial al controlador, haciendo uso del protocolo RS232.

ser = serial.Serial(

port='COM1',

baudrate=9600,

parity=serial.PARITY\_ODD,

stopbits=serial.STOPBITS\_TWO,

bytesize=serial.SEVENBITS

)

ser.isOpen()

x\_min = min(my\_points[:,0])

x\_max = max(my\_points[:,0])

scale = (abs(x\_min)+abs(x\_max))/240

index = 0

inp = 1

while 1 :

j = 0

if index < len(my\_points):

j = 1

inp = str([int(my\_points[index][0]\*scale), int(my\_points[index][1]\*scale)])

if j == 0:

ser.close()

break

else:

inp += '\r\n'

ser.write(inp.encode() )

out = ''

while ser.inWaiting() > 0:

out += ser.read(1)

index += 1

plt.figure()

plt.plot(my\_points[:,0], my\_points[:,1], 'r-')

plt.show()

for i in my\_points:

if len(points) == 1:

break

j = points.index(i)

minimo = min(Distancias[j])

index = Distancias[j].index(minimo)

my\_points.append(points[index])

Distancias.remove(Distancias[j])

for k in range(len(Distancias)):

Distancias[k].remove(Distancias[k][j])

points.remove(points[j])

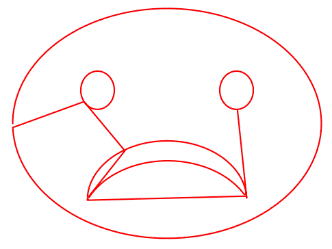
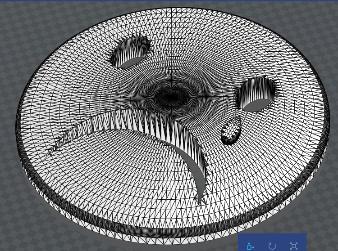
my\_points.append(points[0])

my\_points = np.array(my\_points)

En la siguiente secuencia de imágenes se ilustra mejor el funcionamiento del aplicativo

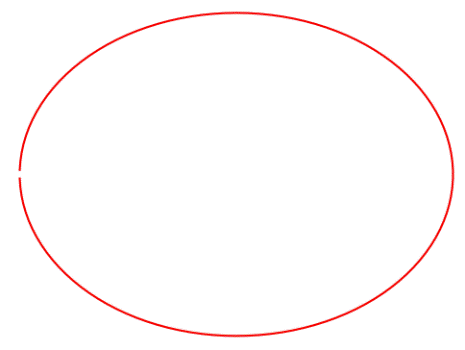
**EJEMPLO 1**

Figura 3D Figura 2D



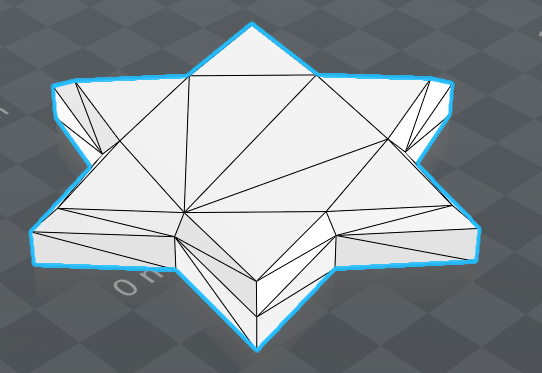
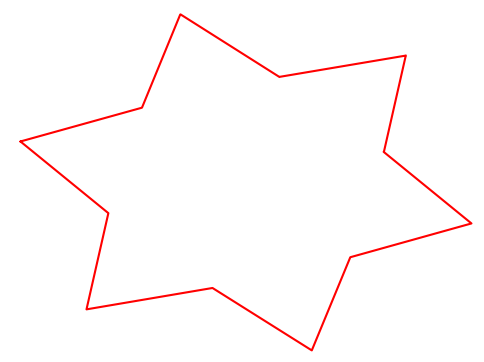
**EJEMPLO 2**

Figura 3D Figura 2D



**EJEMPLO 3**

Figura 3D Figura 2D

**WEBGRAFÍA**

*¿Qué es un fichero STL? - R3ALD. (2017). R3ald.com. Recuperado el 14 de mayo de 2017, de http://www.r3ald.com/que-es-un-fichero-stl*