Universidad Autónoma de Baja California

Facultad De Ciencias Químicas E Ingeniería

Graficación

Metas unidad 4

Soto Elenes Brian Ramiro	#1254563
Amézquita Becerra Carlos Daniel	#1262695
Rivera Soto Karen Dayanara	#1271872

Grupo: 551

Dr. Juan Ramon Castro Rodríguez

Tijuana Baja California a martes, 30 de noviembre del 2021

In [1]:

```python
import numpy as np
import matplotlib.pyplot as plt
import math
```

In [2]:

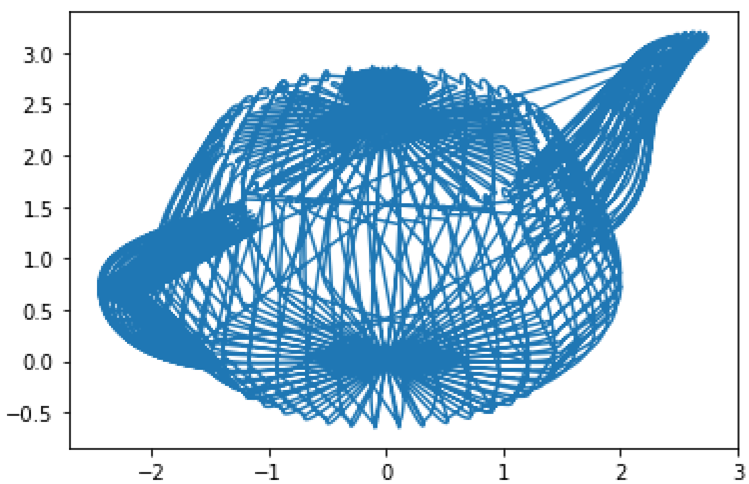
```python
#viewProjMatrix.py
def viewProjMatrix(az, el, phi=0, target=[0,0,0]):
    if phi==0 and target==[0,0,0]:
        phi=0

    if phi!=0 or target!=[0,0,0]:
        if phi>0:
            d=math.sqrt(2)/2/math.tan(phi*math.pi/360)
        else:
            phi=0

    el=((((el+180)%360)+360)%360)-180
    if el>90:
        el=180-el
        az=az+180
    elif el<-90:
        el = -180-el
        az = az + 180

    az = az*math.pi/180
    el = el*math.pi/180

    if target!=[0,0,0]:
        if len(target)!=3:
            print('MATLAB:viewProjMatrix:InvalidInput')
    else:
        target[0] = 0.5 + math.sqrt(3)/2*(math.cos(el)*math.sin(az))
        target[1] = 0.5 + math.sqrt(3)/2*(-math.cos(el)*math.cos(az))
        target[2] = 0.5 + math.sqrt(3)/2*(math.sin(el))

    T = [[1,0,0,-target[0]],[0,1,0,-target[1]],[0,0,1,-target[2]],[0,0,0,1]]

    R = [[math.cos(az),math.sin(az),0,0],
         [-math.sin(el)*math.sin(az),math.sin(el)*math.cos(az),math.cos(el),0],
         [math.cos(el)*math.sin(az),-math.cos(el)*math.cos(az),math.sin(el),0],
         [0,0,0,1]];

    if (phi==0 and target==[0,0,0]) or phi==0:
        M=R
        return M


    Mwc_vc=np.dot(R,T)

    Tpers= [[1,0,0,0],
            [0,1,0,0],
            [0,0,1,0],
            [0,0,-1/d,1]]

    M=np.dot(Tpers,Mwc_vc)
    return M

def paraboloide(x,y):
```

```python
    return (x*x + y*y)

def sec(x):
    return 1.0/math.cos(x*math.pi/180)

def cot(x):
    return 1.0/math.tan(x*math.pi/180)

def csc(x):
    return 1.0/math.sin(x*math.pi/180)
```

In [3]:

```python
#1A
Alpha = -37.5 #ZX
Betha = 30  #ZY
Phi = 0

P = np.loadtxt('teapot_vertex.dat',unpack=True)

M = viewProjMatrix(Alpha,Betha,Phi)
Ph = np.r_[P,[np.ones(len(P[0]))]]
Vh = np.dot(M, Ph)

# Vertices proyectados en el volumen visual 3D (xp,yp,zp)

V = np.delete(Vh, 3, axis=0) / Vh[3]

U = np.delete(V, 2, axis=0)

plt.plot(U[0],U[1])
```

Out[3]: [<matplotlib.lines.Line2D at 0x218eb74e940>]

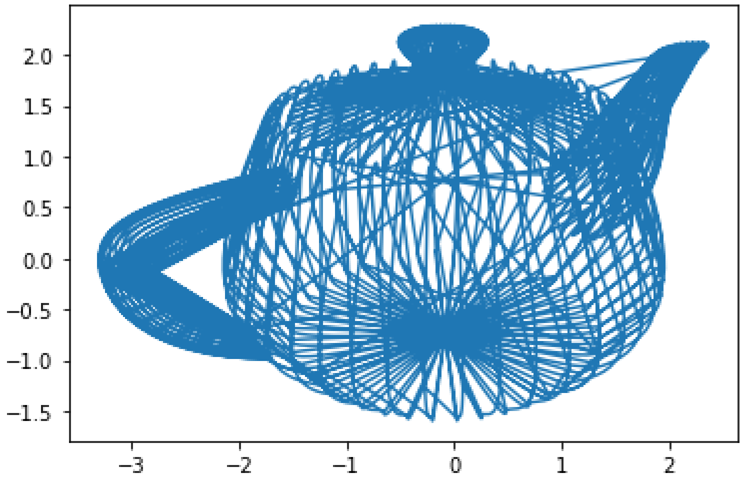


In [4]:

```python
#1B
Alpha = -37.5 #ZX
Betha = 30 #ZY
Phi = 10

P = np.loadtxt('teapot_vertex.dat',unpack=True)

M = viewProjMatrix(Alpha,Betha,Phi)
Ph = np.r_[P,[np.ones(len(P[0]))]]
Vh = np.dot(M, Ph)
```

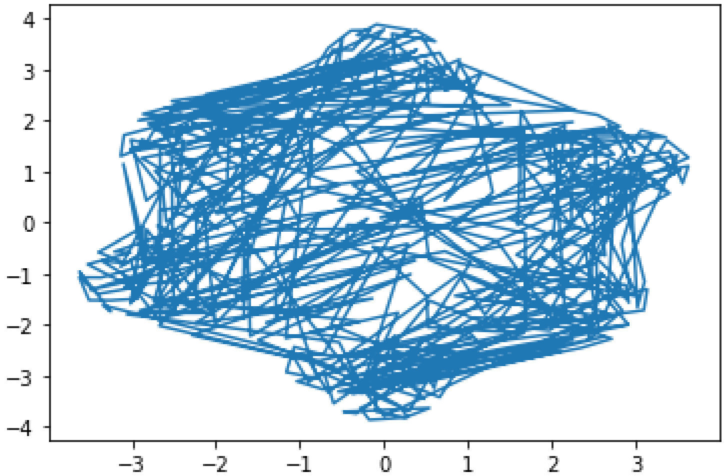
```
# Vertices proyectados en el volumen visual 3D (xp,yp,zp)

V = np.delete(Vh, 3, axis=0) / Vh[3]

U = np.delete(V, 2, axis=0)

plt.plot(U[0],U[1])
```

Out[4]: [<matplotlib.lines.Line2D at 0x218ebecac10>]



In [5]:
```
#2A
Alpha = -37.; #ZX
Betha = 30  #ZY
Phi = 0

P = np.loadtxt('bumpy_vertex.dat',unpack=True)

M = viewProjMatrix(Alpha,Betha,Phi)
Ph = np.r_[P,[np.ones(len(P[0]))]]
Vh = np.dot(M, Ph)

# Vertices proyectados en el volumen visual 3D (xp,yp,zp)

V = np.delete(Vh, 3, axis=0) / Vh[3]

U = np.delete(V, 2, axis=0)

plt.plot(U[0],U[1])
```

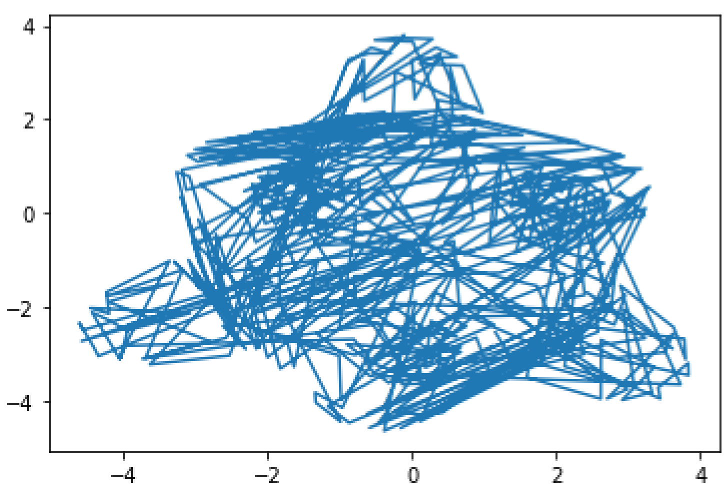
Out[5]: [<matplotlib.lines.Line2D at 0x218ebf89fd0>]



In [6]:

```
#2B
Alpha = -37.5 #ZX
Betha = 30  #ZY
Phi = 10

P = np.loadtxt('bumpy_vertex.dat',unpack=True)

M = viewProjMatrix(Alpha,Betha,Phi)
Ph = np.r_[P,[np.ones(len(P[0]))]]
Vh = np.dot(M, Ph)

# Vertices proyectados en el volumen visual 3D (xp,yp,zp)

V = np.delete(Vh, 3, axis=0) / Vh[3]

U = np.delete(V, 2, axis=0)

plt.plot(U[0],U[1])
```

Out[6]: [<matplotlib.lines.Line2D at 0x218ebfa65e0>]

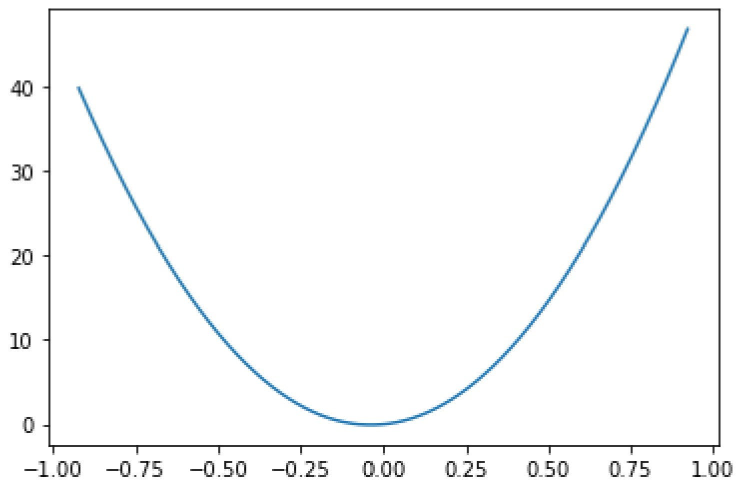


In [7]:

```
#3A
Alpha = -37.5 #ZX
Betha = 30  #ZY
Phi = 0
```

```python
X = np.linspace(-5,5,1000)
Y = np.linspace(-5,5,1000)
Z = paraboloide(X,Y)

P = np.array([X,Y,Z])

M = viewProjMatrix(Alpha,Betha,Phi)
Ph = np.r_[P,[np.ones(len(P[0]))]]
Vh = np.dot(M, Ph)

# Vertices proyectados en el volumen visual 3D (xp,yp,zp)

V = np.delete(Vh, 3, axis=0) / Vh[3]

U = np.delete(V, 2, axis=0)

plt.plot(U[0],U[1])
```

Out[7]: [<matplotlib.lines.Line2D at 0x218ebffd370>]

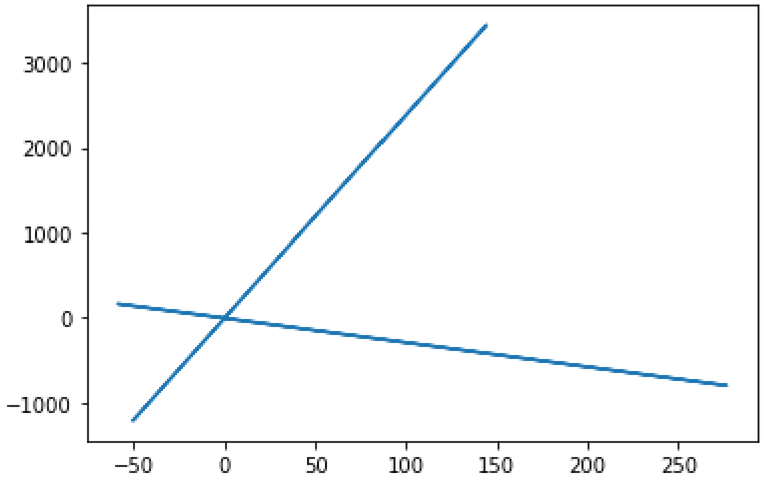


In [8]:

```python
#3B
Alpha = -37.5 #ZX
Betha = 60  #ZY
Phi = 10

X = np.linspace(-5,5,1000)
Y = np.linspace(-5,5,1000)
Z = paraboloide(X,Y)

P = np.array([X,Y,Z])

M = viewProjMatrix(Alpha,Betha,Phi)
Ph = np.r_[P,[np.ones(len(P[0]))]]
Vh = np.dot(M, Ph)

# Vertices proyectados en el volumen visual 3D (xp,yp,zp)

V = np.delete(Vh, 3, axis=0) / Vh[3]

U = np.delete(V, 2, axis=0)

plt.plot(U[0],U[1])
```

```
print(M)
```

```
[[ 0.79335334 -0.60876143  0.         -0.09229596]
 [ 0.52720286  0.68706415  0.5        -0.4241208 ]
 [-0.30438071 -0.39667667  0.8660254  -0.83248401]
 [ 0.03766031  0.04907987 -0.10715129  1.10300129]]
```



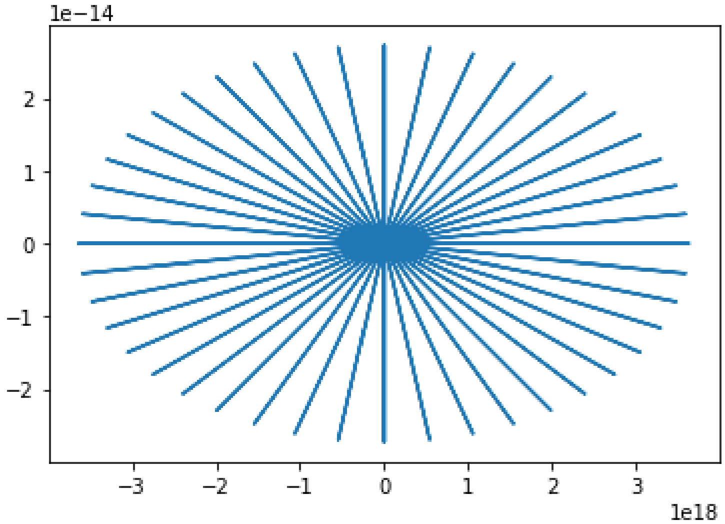
In [9]:

```
def Mnormsymmpers(theta, Znear, Zfar, aspect):
    MNmatrix = [[cot(theta/aspect), 0, 0, 0],
                [0, cot(theta/2), 0, 0],
                [0, 0, (Znear + Zfar)/(Znear - Zfar), (2 * Znear * Zfar)/(Znear - Zfar)
                [0, 0, -1, 0]]
    return MNmatrix
```

In [10]:

```
#4
theta = 180
Znear = 1
Zfar = 10
aspect = 1

P = np.loadtxt('teapot_vertex.dat',unpack=True)

M =  Mnormsymmpers(theta, Znear, Zfar, aspect)
Ph = np.r_[P,[np.ones(len(P[0]))]]
Vh = np.dot(M, Ph)

# Vertices proyectados en el volumen visual 3D (xp,yp,zp)

V = np.delete(Vh, 3, axis=0) / (Vh[3]+1/1000)

U = np.delete(V, 2, axis=0)

plt.plot(U[0],U[1])
```

Out[10]: [<matplotlib.lines.Line2D at 0x218ec138d00>]



In [ ]:

In [ ]: