UNIVERSIDAD NACIONAL DE SAN AGUSTÍN

Facultad de Ingeniería de Producción y Servicios

Ciencia de la Computación

Práctica 08

ALUMNOS:
Pfuturi Huisa, Oscar David
Quispe Menor, Hermogenes
Fernandez Mamani, Brayan Gino
Quiñonez Lopez, Efrain German
Santos Apaza, Yordy Williams

DOCENTE:
MSc. Vicente Machaca Arceda

CURSO:
Computación Gráfica

9 de junio de 2021

Índice

1. Problema 1: Dada la Figura 1, determine las coordenadas de cada vértice del plano rojo, y dibuje el plano.

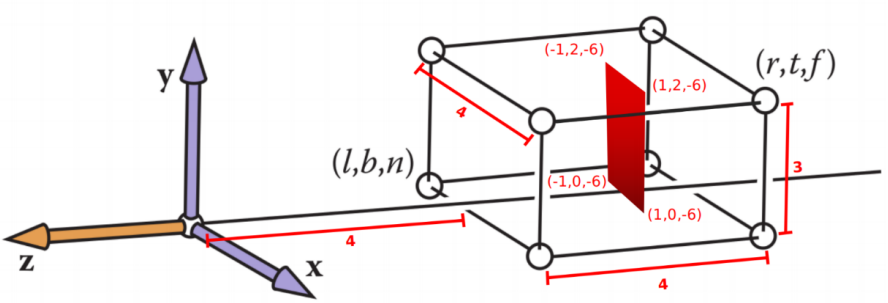

Figura 1: Plano dentro de una vista de volumen ortográfico.

1.1. Considere un monitor de 100 × 80, una proyección ortográfica y una cámara con los siguientes datos:

- e = (0,5,2)
- g = (0,-2,-5)
- t = (0,1,0)

```
import numpy as np

nx=100
ny=80

#Matriz Mvp
Mvp=np.array([
 [nx/2, 0,0,(nx-1)/2],
 [0,ny/2,0,(ny-1)/2],
 [0,0,1,0],
 [0,0,0,1]
 ])

# Puntos del plano rojo
plane = np.array([[-1,1,1,-1],
                  [2,2,0,0],
                  [-6,-6,-6,-6],
                  [1,1,1,1]])

#valores de l,b,n,r,t y f
l=-2; b=0; n=-4
r=2; t=3; f=-8

#Creacion de la matriz ortogonal
Morth=np.array([[2/(r-l),0,0,-((r+l)/(r-l))],
                [0,2/(t-b),0,-((t+b)/(t-b))],
```

```
                    [0,0,2/(n-f),-((n+f)/(n-f))],
                    [0,0,0,1]])
print('Matriz MVP: \n',Mvp)
```

```
Matriz MVP:
 [[50.   0.   0.  49.5]
 [ 0.  40.   0.  39.5]
 [ 0.   0.   1.   0. ]
 [ 0.   0.   0.   1. ]]
```

Figura 2

```
#Proyeccion de transformacion ortografica
M=np.matmul(Mvp,Morth)
Orthographic = np.matmul(M,plane)
print('Proyeccion Ortografica: \n', Orthographic)

e = np.array([0,5,2])
g = np.array([0,-2,-5])
t = np.array([0,1,0])

w = -(g/np.linalg.norm(g))
t_w = np.cross(t,w)
u = t_w / (np.linalg.norm(t_w))
v = np.cross(w,u)

print('vector U: \n',u)
print('vector V: \n',v)
print('vector W: \n',w)

#Construccion de la Mcam
Mcam_ = np.zeros((4,4))
Mcam_[0:3,0] = u
Mcam_[0:3,1] = v
Mcam_[0:3,2] = w
Mcam_[0:3,3] = e
Mcam_[3,3] = 1
#print(Mcam_)
Mcam = np.linalg.inv(Mcam_)
print('matriz Mcam:\n ',Mcam)

#Proyecion de transformacion de camara: Ortografica
M_ = np.matmul(Mvp,Morth)
M =  np.matmul(M_, Mcam)
M_orth_cam = np.matmul(M,plane)
```

```
34 print('Proyecion ortografica con transformacion de camara:\n',
       M_orth_cam)
```

```
Proyecion ortografica con transformacion de camara:
 [[ 24.5         74.5         74.5         24.5       ]
 [  4.45187568   4.45187568 -45.06688116 -45.06688116]
 [ -1.27099278  -1.27099278  -1.64238345  -1.64238345]
 [  1.           1.           1.           1.        ]]
```

Figura 3

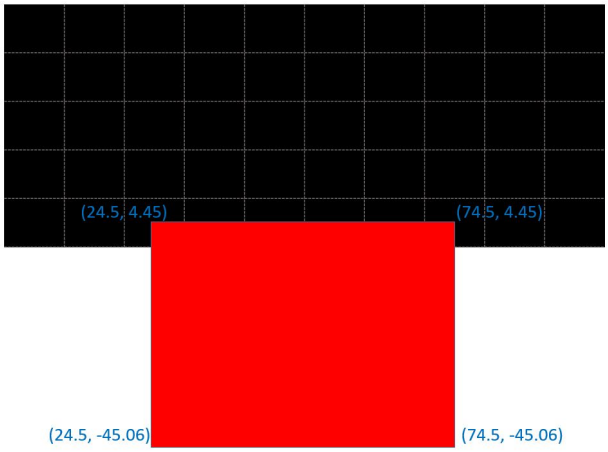

Figura 4: Vista de pantalla

1.2. Realice lo mismo, pero ahora con esta cámara y compare los resultados, debe simular como se graficaría en el monitor.

- e = (0,2,2)
- g = (0,-2,-5)
- t = (0,1,0)

```
1
2 e = np.array([0,2,2])
3 g = np.array([0,-2,-5])
4 t = np.array([0,1,0])
```

```
Proyecion ortografica con transformacion de camara:
 [[24.5         74.5         74.5         24.5       ]
 [78.73001096 78.73001096 29.21125411 29.21125411]
 [-0.71390676 -0.71390676 -1.08529744 -1.08529744]
 [ 1.           1.           1.           1.        ]]
```

Figura 5

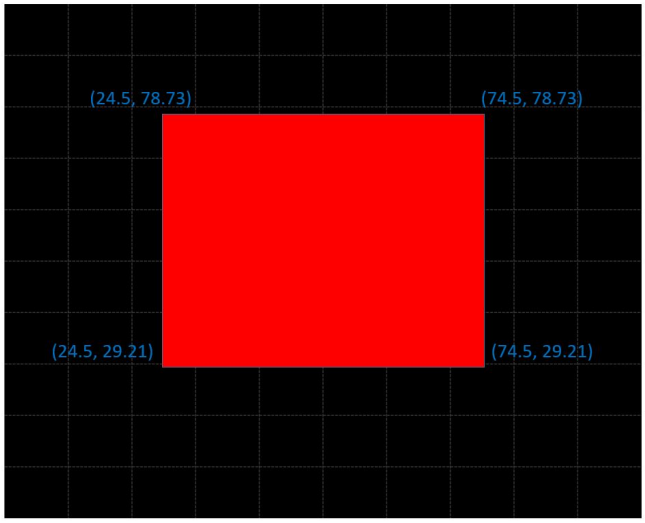

Figura 6: Vista de pantalla

2. Problema 2: Realice el mismo ejercicio anterior, pero ahora considere una proyección perspectiva. Debe utilizar la cámara en las dos posiciones mencionadas anteriormente y luego debe comparar sus resultados, debe simular como se graficaría en el monitor

2.1. a

```
1
2  g = np.array([0,-2,-5])
3  t = np.array([0,1,0])
4  e = np.array([0,5,2])
5
6  #construccion de la matriz perspectiva
7  p = np.array([[n,0,0,0],
8                [0,n,0,0],
9                [0,0,n+f,-n*f],
10               [0,0,1,0]])
11 #proyecion de transformacion perspectiva
12 Mper = np.matmul(Morth,p)
13 print('Matriz perspectica: \n', Mper)
14
15 #Proyecion de transformacion de camara: Perspectiva
16 M_ = np.matmul(Mvp,Mper)
17 M =  np.matmul(M_, Mcam)
18 M_per_cam = np.matmul(M,plane)
19
20 Mcam_ = np.zeros((4,4))
21 Mcam_[0:3,0] = u
22 Mcam_[0:3,1] = v
23 Mcam_[0:3,2] = w
24 Mcam_[0:3,3] = e
```

```
Mcam_[3,3] = 1
#print(Mcam_)
Mcam = np.linalg.inv(Mcam_)
#Convirtiendo a coordenadas cartesianas
M_Homo =  np.zeros((3,4))
M_Car =  np.zeros((3,4))
Homo = np.zeros((1,4))
M_Homo[0:3,0:4] = M_per_cam[0:3,0:4]
Homo[0,0:4] = M_per_cam[3,0:4]
M_Car[0:3,0:2] = M_Homo[0:3,0:2]/Homo[0,0]
M_Car[0:3,2:4] = M_Homo[0:3,2:4]/Homo[0,2]
print('Proyecion perspectiva con transformacion de camara:\n',M_Car)
```

```
Proyecion perspectiva con transformacion de camara:
 [[ 37.79311998  61.20688002  60.27032961  38.72967039]
 [  1.81884058   1.81884058 -19.7        -19.7       ]
 [ -1.1268992   -1.1268992   -1.27674726  -1.27674726]]
```

Figura 7

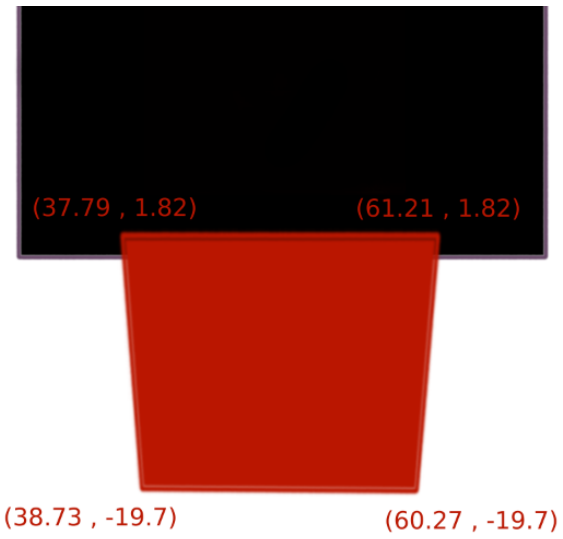

Figura 8: Vista de pantalla

2.2. b

```

e = np.array([0,2,2])
g = np.array([0,-2,-5])
t = np.array([0,1,0])
```

```
Proyecion perspectiva con transformacion de camara:
 [[36.03708798 62.96291202 61.73901093 37.26098907]
 [42.16666667 42.16666667 14.04545455 14.04545455]
 [-0.84593408 -0.84593408 -1.04175825 -1.04175825]]
```

Figura 9

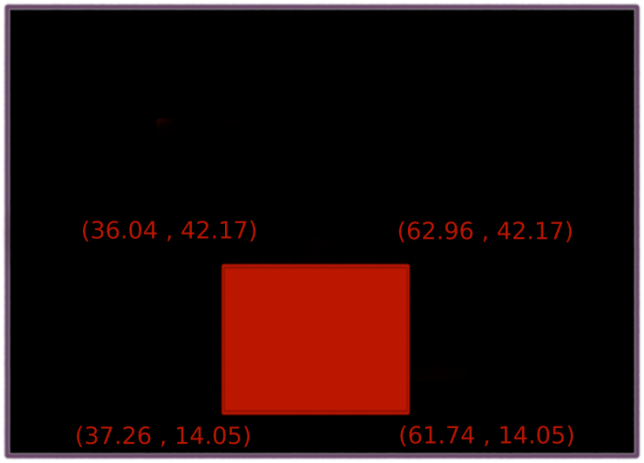

Figura 10: Vista de pantalla

3. Problema 3: Ahora consideramos el concepto de *Field of View*. Para un ángulo θ = 60, halle las coordenadas del plano de la Figura 1, en un monitor de 100 x 80. Utilice una camara con estos datos:

- e = (0,4,2)
- g = (0,-2,-5)
- t = (0,1,0)

```
#Field Of view
n = -4; f = -8
n_x = 100; n_y = 80
theta = np.radians(60)

#t, b, r, l
t = np.tan(theta/2)*np.abs(n)
r = (n_x/n_y)*t
l = -r
b = -t

e = np.array([0,4,2])
g = np.array([0,-2,-5])
t = np.array([0,1,0])

```

```
16 print('valor de t: ',t)
17 print('valor de b: ',b)
18 print('valor de r: ',r)
19 print('valor de l: ',l)
```

```
Field Of View con transformacion ortografica:
 [[32.17949192 66.82050808 66.82050808 32.17949192]
 [58.79802563 58.79802563 26.63464958 26.63464958]
 [-1.08529744 -1.08529744 -1.45668812 -1.45668812]
 [ 1.          1.          1.          1.        ]]
```

Figura 11

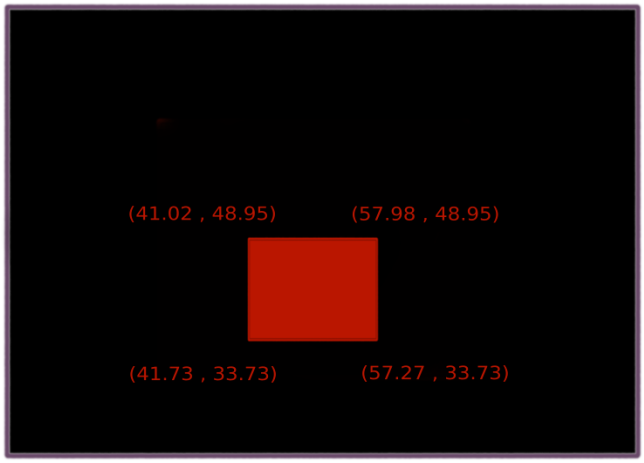

Figura 12: Vista de pantalla

```
Field Of View con transformacion perspectiva:
 [[41.0205645  57.9794355  57.27281588 41.72718412]
 [48.94754986 48.94754986 33.72649731 33.72649731]
 [-1.04175825 -1.04175825 -1.20494506 -1.20494506]]
```

Figura 13

4. Enlace

El código fuente está disponible en:

- https://github.com/oscar-pfuturi-h/Comp-Grafica/tree/main/practica_08