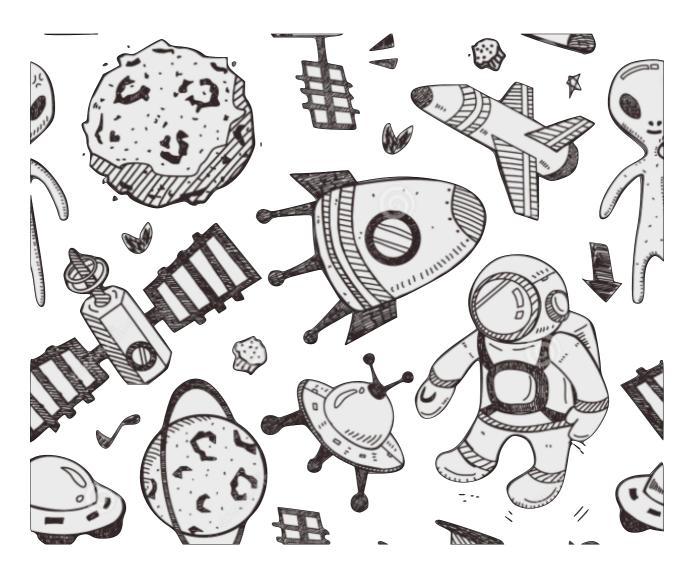
INFORMÁTICA GRÁFICA Y VISUALIZACIÓN



Proyecto Final de Prácticas.

Destrucción de Planetas.

Diseño.

Daniel Moya Leiva. 77360609V.

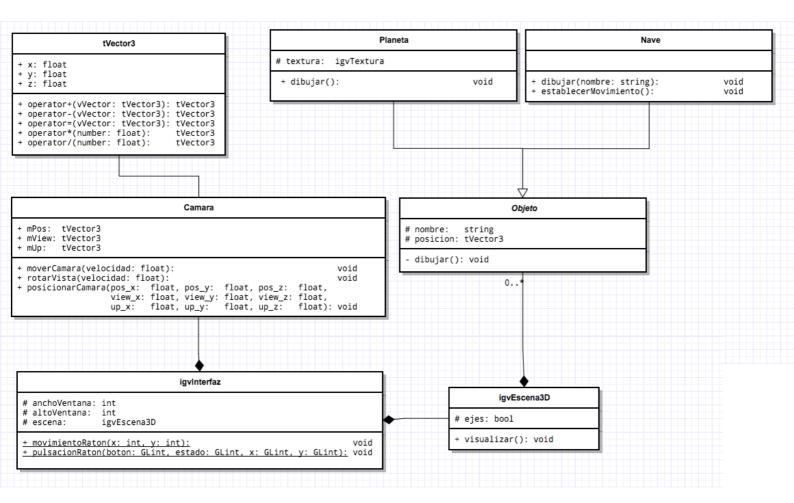
INFORMÁTICA GRÁFICA Y VISUALIZACIÓN

DIAGRAMA UML

Usaremos una clase llamada tVector3 en lugar de igvPunto3D para almacenar posiciones, pues hemos sobrecargado los operadores +, -, =, *, y /. Esto nos da mucha libertad para trabajar con la cámara y las posiciones, y básicamente su funcionalidad es la misma.

Por otro lado, cambiaremos el uso de *igvCamara* por el de *Camara* puesto que tiene una implementación más sencilla de una cámara en primera persona (FPP en adelante). Esta clase tendrá tres métodos básicos: *moverCamara* para cambiar la posición en la que nos encontramos; *rotarVista* para cambiar el campo de visión; y *posicionarCamara* para establecer el inicio de la cámara.

En cuanto a los objetos que habrá por la escena, usaremos una clase abstracta como es *Objeto* de la que heredan las clases *Planeta* y *Satelite* que serán los encargados de mover y dibujar los elementos que están en la escena.



INFORMÁTICA GRÁFICA Y VISUALIZACIÓN GRAFO DE ESCENA. 0.5, 0.5}; 0.0, 0.0}; 0.15, 0.61}; GLfloat gris[] = {0.0, = {0.0, GLfloat negro[] Satélite. GLfloat azul[] GLfloat rojo[] glTranslatef(0, 0, -2);GLUquadricObj *cuerpo; cuerpo = gluNewQuadric(); gluQuadricDrawStyle(cuerpo, GLU_FILL); gluCylinder(cuerpo, 2, 2, 4, 20, 20); glMaterialfv(GL_FRONT, GL_EMISSION, negro); glPushMatrix(); glPushMatrix(); glTranslatef(0, 0, 4); glPushMatrix(); // Cabeza // Brazo glutSolidSphere(2, 50, 50); glPopMatrix(); glTranslatef(6, 0, 2); glRotatef(-90, 0, 1, 0); GLUquadricObj *cabeza; cabeza = gluNewQuadric(); GLUquadricObj *brazoder; brazoder = gluNewQuadric(); gluQuadricDrawStyle(cabeza, GLU_FILL); gluCylinder(cabeza, 0.25, 2, 4, 20, 20); gluQuadricDrawStyle(brazoder, GLU_FILL); gluCylinder(brazoder, 0.5, 0.5, 4, 20, 20); // Antena glTranslatef(0, 0, -2); GLUquadricObj *antena; antena = gluNewQuadric(); glMaterialfv(GL_FRONT, GL_EMISSION, azul); glScalef(2, 2, 0.1); glutSolidCube(5); glMaterialfv(GL_FRONT, GL_EMISSION, rojo); gluQuadricDrawStyle(antena, GLU_FILL); gluCylinder(antena, 0.25, 0.25, 2, 20, 20); glutSolidSphere(0.5, 50, 50); lPopMatrix();

Implementación Satélite.

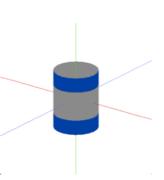
```
glPushMatrix();
                                                      // Brazo Izquierdo
                                                      glMaterialfv(GL_FRONT, GL_EMISSION, negro);
  glMaterialfv(GL_FRONT, GL_EMISSION, gris);
                                                      glPushMatrix();
  // Cuerpo
                                                        // Brazo
  glTranslatef(0, 0, -2);
                                                        glTranslatef(-6, 0, 2);
  GLUquadricObj *cuerpo;
                                                        glRotatef(90, 0, 1, 0);
  cuerpo = gluNewQuadric();
                                                        GLUquadricObj *brazoizq;
  gluQuadricDrawStyle(cuerpo, GLU_FILL);
                                                        brazoizq = gluNewQuadric();
  gluCylinder(cuerpo, 2, 2, 4, 20, 20);
                                                        gluQuadricDrawStyle(brazoizq, GLU_FILL);
                                                        gluCylinder(brazoizq, 0.5, 0.5, 4, 20, 20);
                                                        // Panel
  // Parte trasera
                                                        glMaterialfv(GL_FRONT, GL_EMISSION, azul);
  glPushMatrix();
                                                        glScalef(2, 2, 0.1);
    glTranslatef(0, 0, 4);
                                                        glutSolidCube(5);
    glutSolidSphere(2, 50, 50);
                                                      glPopMatrix();
  glPopMatrix();
                                                      // Brazo Derecho
  // Parte delantera
                                                      glMaterialfv(GL_FRONT, GL_EMISSION, negro);
  glPushMatrix();
                                                      glPushMatrix();
    // Cabeza
                                                        // Brazo
    glTranslatef(0, 0, -4);
                                                        glTranslatef(6, 0, 2);
    GLUquadricObj *cabeza;
                                                        glRotatef(-90, 0, 1, 0);
                                                        GLUquadricObj *brazoder;
   cabeza = gluNewQuadric();
    gluQuadricDrawStyle(cabeza, GLU_FILL);
                                                        brazoder = gluNewQuadric();
    gluCylinder(cabeza, 0.25, 2, 4, 20, 20);
                                                        gluQuadricDrawStyle(brazoder, GLU_FILL);
                                                        gluCylinder(brazoder, 0.5, 0.5, 4, 20, 20);
    // Antena
    glTranslatef(0, 0, -2);
                                                        // Panel
                                                        glMaterialfv(GL_FRONT, GL_EMISSION, azul);
    GLUquadricObj *antena;
    antena = gluNewQuadric();
                                                        glScalef(2, 2, 0.1);
    glMaterialfv(GL_FRONT, GL_EMISSION, rojo);
                                                        glutSolidCube(5);
    gluQuadricDrawStyle(antena, GLU_FILL);
                                                      glPopMatrix();
    gluCylinder(antena, 0.25, 0.25, 2, 20, 20);
    glutSolidSphere(0.5, 50, 50);
                                                   glPopMatrix();
  glPopMatrix();
```

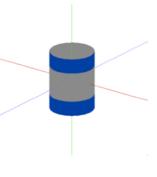
GRAFO DE ESCENA.

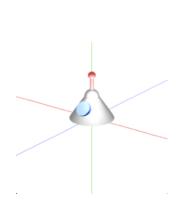
Cohete.

```
GLfloat gris[]
GLfloat rojo[]
GLfloat azul[]
                                            = {0.5,
= {0.8,
= {0.0,
= {0.4,
                                                                   0.5, 0.5};
0.15, 0.15};
0.15, 0.61};
                                                                    0.15,
 GLfloat azulClaro[]
```

```
// Cuerpo
glMaterialfv(GL_FRONT, GL_EMISSION, gris);
glTranslatef(0, 0, -4);
GLUquadricObj *cuerpo;
cuerpo = gluNewQuadric();
gluQuadricDrawStyle(cuerpo, GLU_FILL);
gluCylinder(cuerpo, 3, 3, 8, 20, 20);
glPushMatrix();
   glMaterialfv(GL_FRONT, GL_EMISSION, azul);
  gluCylinder(cuerpo, 3.01, 3.01, 2, 20, 20);
 glTranslatef(0, 0, 6);
gluCylinder(cuerpo, 3.01, 3.01, 2, 20, 20);
glPopMatrix();
```

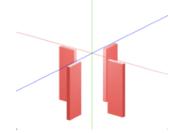






```
/ Cabeza
|/ Cabeza
| Materialfv(GL_FRONT, GL_EMISSION, gris);
| PushMatrix();
| glTranslatef(0, 0, -3);
 GLUquadricObj *cabeza;
cabeza = gluNewQuadric();
 gluQuadricDrawStyle(cabeza, GLU_FILL);
gluCylinder(cabeza, 1, 3, 3, 20, 20);
// Escotilla
glPushMatrix();
glMaterialfv(GL_FRONT, GL_EMISSION, azul);
glRotatef(-125, 1, 0, 0);
glTranslatef(9, -2, 0);
GLUquadricObj *escotilla;
escotilla = gluNewQuadric();
      gluQuadricDrawStyle(escotilla, GLU_FILL);
gluCylinder(escotilla, 1, 1, 1, 20, 20);
 // Ventana
glMaterialfv(GL_FRONT, GL_EMISSION, azulClaro);
glTranslatef(0, 0, 1);
      GLUquadricObj *ventana;
ventana = gluNewQuadric();
 gluQuadricDrawStyle(ventana, GLU_FILL);
gluDisk(ventana, 0, 1, 50, 50);
glPopMatrix();
 // Antena
glPushMatrix();
glMaterialfv(GL_FRONT, GL_EMISSION, gris);
glutSolidSphere(1, 50, 50);
     glMaterialfv(GL_FRONT, GL_EMISSION, rojo);
GLUquadricObj *antena;
antena = gluNewQuadric();
      gluQuadricDrawStyle(antena, GLU_FILL);
gluCylinder(antena, 0.25, 0.25, 2, 20, 20);
  glutSolidSphere(0.5, 50, 50);
glPopMatrix();
PopMatrix();
```





Implementación Cohete.

```
glPushMatrix();
                                               // Cabeza
  glScalef(0.5, 0.5, 0.5);
                                               glMaterialfv(GL_FRONT, GL_EMISSION, gris);
  glRotatef(90, 1, 0, 0);
                                               glPushMatrix();
// Cuerpo
                                                 glTranslatef(0, 0, -3);
glMaterialfv(GL_FRONT, GL_EMISSION, gris);
                                                 GLUquadricObj *cabeza;
glTranslatef(0, 0, -4);
                                                 cabeza = gluNewQuadric();
GLUquadricObj *cuerpo;
                                                 gluQuadricDrawStyle(cabeza, GLU_FILL);
cuerpo = gluNewQuadric();
                                                 gluCylinder(cabeza, 1, 3, 3, 20, 20);
gluQuadricDrawStyle(cuerpo, GLU_FILL);
gluCylinder(cuerpo, 3, 3, 8, 20, 20);
                                                 // Escotilla
glPushMatrix();
                                                 glPushMatrix();
   glMaterialfv(GL_FRONT, GL_EMISSION, azul);
                                                   glMaterialfv(GL_FRONT, GL_EMISSION, azul);
   gluCylinder(cuerpo, 3.01, 3.01, 2, 20, 20);
                                                   glRotatef(-125, 1, 0, 0);
   glTranslatef(0, 0, 6);
                                                   glTranslatef(0, -2, 0);
   gluCylinder(cuerpo, 3.01, 3.01, 2, 20, 20);
                                                   GLUquadricObj *escotilla;
glPopMatrix();
                                                   escotilla = gluNewQuadric();
                                                   gluQuadricDrawStyle(escotilla, GLU_FILL);
// Alas laterales
                                                   gluCylinder(escotilla, 1, 1, 1, 20, 20);
glPushMatrix();
   glMaterialfv(GL_FRONT, GL_EMISSION, rojo);
                                                   // Ventana
   glRotatef(90, 1, 0, 0);
                                                   glMaterialfv(GL_FRONT, GL_EMISSION, azulClaro);
  for (int i = -4; i \le 4; i+=8) {
                                                   glTranslatef(0, 0, 1);
    glPushMatrix();
                                                   GLUquadricObj *ventana;
       glTranslatef(i, 8, 0);
                                                   ventana = gluNewQuadric();
       glScalef(0.5, 1.5, 0.1);
                                                   gluQuadricDrawStyle(ventana, GLU_FILL);
      glutSolidCube(5);
                                                   gluDisk(ventana, 0, 1, 50, 50);
    glPopMatrix();
                                                 glPopMatrix();
   }
                                                 // Antena
   // Alas frontales
                                                 glPushMatrix();
   glMaterialfv(GL_FRONT, GL_EMISSION, rojo);
                                                   glMaterialfv(GL_FRONT, GL_EMISSION, gris);
   glRotatef(90, 0, 1, 0);
                                                   glutSolidSphere(1, 50, 50);
   for (int i = 4; i >= -4; i-=8) {
                                                   glTranslatef(0, 0, -3);
    glPushMatrix();
                                                   glMaterialfv(GL_FRONT, GL_EMISSION, rojo);
       glTranslatef(i, 8, 0);
                                                   GLUquadricObj *antena;
       glScalef(0.5, 1.5, 0.1);
                                                   antena = gluNewQuadric();
       glutSolidCube(5);
                                                   gluQuadricDrawStyle(antena, GLU_FILL);
    glPopMatrix();
                                                   gluCylinder(antena, 0.25, 0.25, 2, 20, 20);
                                                   glutSolidSphere(0.5, 50, 50);
glPopMatrix();
                                                 glPopMatrix();
                                               glPopMatrix();
```