



## UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

## FACULTAD DE INGENIERÍA

**Asignatura:** Computación Gráfica e Interacción Humano Computadora

**Grupo:** 5

**Semestre:** 2022-2

### **Manual de Desarrollo**

**Fecha Límite de Entrega:** 26/05/2022

**Profesor:** José Roque Román Guadarrama

**Alumnos:**

- Colin Santos Luis Froylan
- Najera Noyola Karla Andrea

## ¿Qué es este manual? / What is this manual?

En este manual se describe la realización del proyecto con base en los recursos empleados, la lógica, la programación del software y los rubros solicitados para este proyecto, especialmente para el rubro de “Elementos a incluir dentro del escenario”.

Este manual no está hecho para dar información acerca de cómo usar el programa (eso se incluye en el manual de usuario) ni está pensado para ayudar a configurar el entorno de compilación (eso se incluye dentro del manual de configuración).

**This manual describes the realization of the project based on the resources used, the logic behind it, the programming process of this software and the rubrics requested for this project, especially for the "Elementos a incluir dentro del escenario" item.**

**This manual is not intended to give information about how to use the program (that can be read from the user manual) nor is it intended to help configure the build environment (that is included in the configuration manual).**

## Información del software / Software Information

Tenemos distintos rubros a cubrir dentro de la categoría de “Elementos a incluir dentro del escenario”, que son los siguientes:

- Geometría
- Avatar
- Recorrido
- Iluminación
- Animación
- Elemento propio
- Audio

A continuación presentamos cada rubro con sus soluciones dentro de nuestro proyecto.

**We have different items to cover within the category of "Elements to include in the scenario", which are the following:**

- Geometry
- Avatar
- Route
- Lightning
- Animation
- Own element
- Audio

**Below we present each rubric item with its solutions within our project.**

## Geometría y Avatar / Geometry and Avatar

El rubro Geometría toma en cuenta a los modelos visibles en el escenario y la texturización de los mismos. El rubro Avatar toma en cuenta que los avatares propuestos (Morgana y Futaba, para este caso) están creados de manera jerárquica.

**The Geometry category takes into account the visible models on the stage and their textures. The Avatar category takes into account that the proposed avatars (Morgana and Futaba, in this case) are created in a hierarchical way.**

Para nuestro caso, todos los modelos (salvo el piso) fueron hechos por nosotros, incluidas las texturas, salvo contadas excepciones.

**In our case, all the models (except the floor) were made by us, even the textures, with a few exceptions.**

Entre las excepciones, encontramos el skybox azul, que nos fue otorgado por el profesor Sergio Valencia en el laboratorio de la materia, solo que nosotros modificamos un poco su apariencia para que vaya acorde al voxel-art, y lo mismo podemos decir del modelo del piso, que nos fue otorgado por el profesor Sergio, solo que nosotros pusimos una textura hecha desde cero. Igualmente, el skybox naranja fue tomado del siguiente link <https://www.pngwing.com/es/free-png-pplmt> en donde no se especifica un autor, aunque igualmente lo modificamos para ponerle un tono más naranja.

**Among the exceptions, there is the blue skybox, which was given to us by Professor Sergio Valencia in the subject laboratory, but it is worth mentioning that we modified its appearance a little so that it is consistent with the voxel-art style, and the same can be said of the model of the floor, which was given to us by Professor Sergio, but we put a texture made from scratch. Likewise, the orange skybox was taken from the following link <https://www.pngwing.com/es/free-png-pplmt> where an author is not specified, although we also modified it to give it a more orange color tone.**

Todos los demás modelos son nuestra propia creación, con ayuda del software MagicaVoxel, que nos permite modelar en voxel-art, exportar a .obj nuestros modelos y también obtener sus materiales y texturas, lo que facilitó mucho esta parte del desarrollo. Lamentablemente, en MagicaVoxel, para modelos pequeños, suele haber problemas con el origen del modelo, lo que se resolvió utilizando el software de 3ds Max 2023.

**All the other models are our own creation, with the help of the MagicaVoxel tool, which allows us to model in voxel-art, export our models to .obj and also obtain their materials and textures, which made this part of the development a lot easier. Unfortunately in MagicaVoxel, for small models, there are issues with the model origin, but all of these issues were solved using the 3ds Max 2023 software.**



Accesos directos a los programas MagicaVoxel y 3ds Max 2023.

#### **Shortcuts to MagicaVoxel and 3ds Max 2023 programs.**

Una vez exportados y corregidos nuestros modelos, fue posible incorporarlos al proyecto. Todos los modelos se encuentran dentro de la carpeta del proyecto “resources < objects”, aunque todos se encuentran en carpetas distintas.

**Once our models were exported and their origins were corrected, it was possible to incorporate them into the project. All the models are located inside the project folder “resources < objects”, although they are all in different folders.**

Este equipo > Data (D:) > Descargas > Octavo Semestre > Computacion Grafica > ProyFinal > ProyectoCG > ProyectoCG > resources > objects >			
Nombre	Fecha de modificación	Tipo	Tamaño
ann	10/05/2022 11:58 a. m.	Carpeta de archivos	
camion_gato	10/05/2022 09:22 a. m.	Carpeta de archivos	
contenedor_basura	08/05/2022 06:34 p. m.	Carpeta de archivos	
estacion	09/05/2022 10:06 a. m.	Carpeta de archivos	
faro_ilum	22/05/2022 06:48 p. m.	Carpeta de archivos	
flora	10/05/2022 11:29 a. m.	Carpeta de archivos	
Futaba	11/05/2022 08:37 a. m.	Carpeta de archivos	
globo_dialogo	11/05/2022 05:01 p. m.	Carpeta de archivos	
hachiko	08/05/2022 06:35 p. m.	Carpeta de archivos	
joker	09/05/2022 10:17 a. m.	Carpeta de archivos	
magnet	09/05/2022 09:19 p. m.	Carpeta de archivos	
morgana	09/05/2022 11:50 a. m.	Carpeta de archivos	
novio_joker	08/05/2022 07:34 p. m.	Carpeta de archivos	
ovni	08/05/2022 06:26 p. m.	Carpeta de archivos	
piramide	09/05/2022 10:19 a. m.	Carpeta de archivos	
piso	10/05/2022 08:31 p. m.	Carpeta de archivos	
pisofail	08/05/2022 06:17 p. m.	Carpeta de archivos	
prueba	10/05/2022 08:31 p. m.	Carpeta de archivos	
qfront	09/05/2022 08:50 p. m.	Carpeta de archivos	

Vistazo a la carpeta “resources < objects”.

#### **A look at the “resources < objects” folder.**

Algunos modelos que hacen uso de jerarquía se encuentran agrupados en carpetas. Por ejemplo, el modelo camion\_gato cuenta con la carrocería y ruedas unidas por jerarquía, y dentro de su carpeta podemos encontrar ambos modelos. Esto pasa también con todos los modelos humanoides, que cuentan con jerarquía en brazos, piernas y cabeza.

**Some models that make use of hierarchy are grouped together in folders. For example, the camion\_gato model has the bodywork and wheels linked by hierarchy, and within its folder we can find both models (car and wheels). This also happens with all humanoid models, which have a hierarchy in arms, legs and head.**

« Data (D) > Descargas > Octavo Semestre > Computacion Grafica > ProyFinal > ProyectoCG > ProyectoCG > resources > objects > camion_gato				
Nombre	Fecha de modificación	Tipo	Tamaño	
llantita_no_llonja.mtl	10/05/2022 09:24 a. m.	Archivo MTL	1 KB	
llantita_no_llonja.obj	10/05/2022 09:21 a. m.	Object File	9 KB	
llantita_no_llonja.png	10/05/2022 08:48 a. m.	Archivo PNG	1 KB	
magiccatbus.mtl	09/05/2022 10:39 a. m.	Archivo MTL	1 KB	
magiccatbus.obj	09/05/2022 10:39 a. m.	Object File	162 KB	
magiccatbus.png	09/05/2022 10:39 a. m.	Archivo PNG	1 KB	

Vistazo a la carpeta camion\_gato, que cuenta con 2 modelos: Carrocería y llantas.

**A look at the camion\_gato folder, which has 2 models: Bodywork and wheels.**

Este equipo > Data (D) > Descargas > Octavo Semestre > Computacion Grafica > ProyFinal > ProyectoCG > ProyectoCG > resources > objects > joker				
Nombre	Fecha de modificación	Tipo	Tamaño	
brazo.mtl	10/05/2022 12:13 p. m.	Archivo MTL	1 KB	
brazo.obj	10/05/2022 12:02 p. m.	Object File	2 KB	
brazo.png	08/05/2022 07:21 p. m.	Archivo PNG	1 KB	
cabeza.mtl	10/05/2022 12:13 p. m.	Archivo MTL	1 KB	
cabeza.obj	10/05/2022 12:05 p. m.	Object File	60 KB	
cabeza.png	09/05/2022 10:16 a. m.	Archivo PNG	1 KB	
joker_completo.mtl	09/05/2022 10:17 a. m.	Archivo MTL	1 KB	
joker_completo.obj	09/05/2022 10:17 a. m.	Object File	117 KB	
joker_completo.png	09/05/2022 10:17 a. m.	Archivo PNG	1 KB	
pierna.mtl	10/05/2022 12:13 p. m.	Archivo MTL	1 KB	
pierna.obj	10/05/2022 12:06 p. m.	Object File	15 KB	
pierna.png	08/05/2022 07:21 p. m.	Archivo PNG	1 KB	
torso.mtl	10/05/2022 12:13 p. m.	Archivo MTL	1 KB	
torso.obj	10/05/2022 12:08 p. m.	Object File	9 KB	
torso.png	08/05/2022 07:20 p. m.	Archivo PNG	1 KB	

Vistazo a la carpeta Joker, que cuenta con varios modelos que componen al modelo principal y se unen por jerarquía.

**A look at the Joker folder, which has several models that make up the main model and are linked together by hierarchy.**

La manera de cargar todos los modelos dentro del proyecto se describe a continuación.

Necesitamos agregar a las bibliotecas necesarias, tanto para realizar operaciones matemáticas como para poder utilizar modelos y texturas.

**The way to load the models within the project is described below.**

**We need to add the correct libraries, both to perform mathematical operations and to be able to use models and textures.**

```
11   #include <glad/glad.h>
12   #include <glfw3.h> //Main
13   #include <stdlib.h>
14   #include <glm/glm.hpp> //Camara y Model
15   #include <glm/gtc/matrix_transform.hpp> //Camara y Model
16   #include <glm/gtc/type_ptr.hpp>
17   #include <time.h>

25   #include <shader_m.h>
26   #include <camera.h>
27   #include <modelAnim.h>
28   #include <model.h>
29   #include <Skybox.h>
30   #include <iostream>
```

Los skyboxes hacen uso de un shader, por lo que es necesario inicializar ese shader.

**Skyboxes make use of a shader, so it is necessary to initialize that shader.**

```
1055     || Shader skyboxShader("Shaders/skybox.vs", "Shaders/skybox.fs");
```

Una vez iniciado el shader, procedemos a cargar las texturas de nuestros skyboxes y a crear los objetos de skybox.

**Once the shader is initialized, we proceed to load our skybox textures and create the skybox objects.**

```

1059     //Se cargan recursos del skybox
1060     vector<std::string> faces
1061     {
1062         "resources/skybox/right.jpg",
1063         "resources/skybox/left.jpg",
1064         "resources/skybox/top.jpg",
1065         "resources/skybox/bottom.jpg",
1066         "resources/skybox/front.jpg",
1067         "resources/skybox/back.jpg"
1068     };
1069
1070     vector<std::string> facesalt
1071     {
1072         "resources/skybox/alt/right.png",
1073         "resources/skybox/alt/left.png",
1074         "resources/skybox/alt/top.png",
1075         "resources/skybox/alt/bottom.png",
1076         "resources/skybox/alt/front.png",
1077         "resources/skybox/alt/back.png"
1078     };
1079
1080     Skybox skybox1 = Skybox(faces);
1081     Skybox skybox2 = Skybox(facesalt);

```

Después, procedemos a usar los skyboxes.

**Next, we proceed to use the skyboxes.**

```

1086     // Shader configuration
1087     skyboxShader.use();
1088     skyboxShader.setInt("skybox1", 0);

```

Una vez en uso, se procede a dibujar el skybox.

**Once in use, we proceed to draw the skybox.**

```

2351     // Se dibuja skybox
2352     skyboxShader.use();
2353     if(skyboxtipe)
2354         skybox1.Draw(skyboxShader, view, projection, camera);
2355     else
2356         skybox2.Draw(skyboxShader, view, projection, camera);

```

Si queremos cambiar el skybox en tiempo de ejecución, simplemente se presiona la tecla N para dibujar al otro skybox.

**If we want to change the skybox at runtime, we simply press the N key to draw the other skybox.**

```

2400     //Skybox
2401     if (glfwGetKey(window, GLFW_KEY_N) == GLFW_PRESS)
2402         skyboxtipe ^= true;

```

Si cerramos el programa, necesitamos cerrar los skyboxes para que dejen de ocupar memoria.

If we close the program, we close the skyboxes so that they stop taking up memory.

```
2375      if (skyboxtype)
2376          skybox1.Terminate();
2377      else
2378          skybox2.Terminate();
```

Para cargar a los modelos, necesitamos cargar cada .obj, tener la textura y su material en la misma carpeta para que se muestren correctamente. Posteriormente, podemos empezar la carga en OpenGL.

To load the models, we need to load each .obj, have the texture and its material in the same folder for them to display correctly. Later, we can start the OpenGL load of models.

```
1090      // Carga de modelos
1091      // Edificios
1092      Model piso("resources/objects/piso/piso.obj");
1093      Model qfront("resources/objects/qfront/qfront_chido.obj");
1094      Model magnet("resources/objects/magnet/magnet.obj");
1095      Model torniquetes("resources/objects/estacion/torniquetes.obj");
1096      Model estacion("resources/objects/estacion/estacion.obj");
1097      Model espera("resources/objects/estacion/espera_trenes.obj");
1098      Model piramide("resources/objects/piramide/piramide.obj");
1099      Model hachiko("resources/objects/hachiko/buchiko_estatua.obj");
1100
1101      //Vehiculos
1102      Model ovni("resources/objects/ovni/ovni.obj");
1103      Model vagón("resources/objects/tren/tren.obj");
1104      Model cabina("resources/objects/tren/cabina.obj");
1105      Model camion("resources/objects/camion_gato/magiccatbus.obj");
1106      Model rueda("resources/objects/camion_gato/llantita_no_lonja.obj");
1107
1108      //Flora
1109      Model arbol("resources/objects/flora/green_tree.obj");
1110      Model arbusto("resources/objects/flora/arbusto.obj");
1111      Model circulo("resources/objects/flora/circulo-para-arbol.obj");
1112      Model planta("resources/objects/flora/planta_amarilla.obj");
1113
1114      //Reloj
1115      Model minutos("resources/objects/reloj/manecilla_minutos.obj");
1116      Model horas("resources/objects/reloj/manecilla_horas.obj");
```

En esta parte, la jerarquía realmente no importa, solo importa cargar los modelos en cualquier orden, aunque sí organizamos la carga con cierto orden para no confundirnos.

In this part, the hierarchy order doesn't really matter, it only matters to load the models, although we do organize the loading in a certain order to avoid confusion.

```

1118 //Morgana
1119 Model cabezaMorgana("resources/objects/morgana/cabeza.obj");
1120 Model torsoMorgana("resources/objects/morgana/torso.obj");
1121 Model brazoMorgana("resources/objects/morgana/brazo_completo.obj");
1122 Model piernaMorgana("resources/objects/morgana/pierna.obj");
1123 Model patasCorriendoMorgana("resources/objects/morgana/patas_correr.obj");
1124
1125 //Joker
1126 Model cabezaJoker("resources/objects/joker/cabeza.obj");
1127 Model torsoJoker("resources/objects/joker/torso.obj");
1128 Model brazoJoker("resources/objects/joker/brazo.obj");
1129 Model piernaJoker("resources/objects/joker/pierna.obj");
1130
1131 //Novio Joker
1132 Model cabezaAkechi("resources/objects/novio_joker/cabeza.obj");
1133 Model torsoAkechi("resources/objects/novio_joker/torso.obj");
1134 Model brazoAkechi("resources/objects/novio_joker/brazo.obj");
1135 Model piernaAkechi("resources/objects/novio_joker/pierna.obj");
1136
1137 //Ann
1138 Model cabezaAnn("resources/objects/ann/cabeza.obj");
1139 Model torsoAnn("resources/objects/ann/torso.obj");
1140 Model brazoAnn("resources/objects/ann/brazo.obj");
1141 Model piernaAnn("resources/objects/ann/pierna1.obj");
1142 Model pierna2Ann("resources/objects/ann/pierna2.obj");

```

Para el caso de Futaba, como no usamos un .obj, sino un .dae, inicializamos un shader de animación, pues sus animaciones por defecto fueron obtenidas por medio de Mixamo. Como son 2 animaciones, entonces cargamos 2 modelos con animación.

**In the case of Futaba, since we are not using an .obj, but a .dae, we initialize an animation shader, because its default animations were obtained through Mixamo. Since there are 2 animations for Futaba, then we load the 2 models with animation.**

```

1144 //Futaba 1 (Flotando)
1145 ModelAnim Futaba1("resources/objects/Futaba/Floating/Floating.dae");
1146 Futaba1.initShaders(animShader.ID);
1147
1148 //Futaba 2 (Gritando)
1149 ModelAnim Futaba2("resources/objects/Futaba/Yelling/Yelling.dae");
1150 Futaba2.initShaders(animShader.ID);
1151
1152 //Globo de dialogo
1153 Model globo("resources/objects/globo_dialogo/globo_con_dialogo.obj");
1154
1155 //Vía del tren
1156 Model via("resources/objects/via/via.obj");
1157
1158 //Faro de iluminación spotlight
1159 Model faro("resources/objects/faro_ilum/faro.obj");

```

El siguiente paso después de la carga es dibujarlos, pero para eso haces uso de un shader conocido como staticShader, mientras que para modelos con animación usamos el animShader. Necesitamos usarlos antes de dibujar los modelos. El shader static también se usa para la iluminación. Una vez

que se inician los shaders, procedemos a aplicar transformaciones geométricas, matrices auxiliares para implementación de jerarquía y finalmente dibujar. Debido a la repetición de algunos modelos, preferimos no mostrar todo el código de dibujo, sino partes del mismo.

The next step after model loading is to draw the loaded models, but for that we make use of a shader known as staticShader, while for animated models we use the animShader. We need to use them before drawing the models. The static shader is also used for lighting. Once the shaders are started, we proceed to apply geometric transformations, auxiliary matrices for hierarchy implementation and finally draw all the models needed. Due to the constant repetition of some models, we prefer not to show all the drawing code, but parts of it.

1228	staticShader.use();
------	---------------------

```
1304 // Personajes animados
1305 animShader.use();
1306 animShader.setMat4("projection", projection);
1307 animShader.setMat4("view", view);
1308
1309 animShader.setVec3("material.specular", glm::vec3(0.5f));
1310 animShader.setFloat("material.shininess", 32.0f);
1311 animShader.setVec3("light.ambient", ambientColor);
1312 animShader.setVec3("light.diffuse", diffuseColor);
1313 animShader.setVec3("light.specular", 1.0f, 1.0f, 1.0f);
1314 animShader.setVec3("light.direction", lightDirection);
1315 animShader.setVec3("viewPos", camera.Position);
1316
1317 // Dibujo Futaba 1
1318 model = glm::translate(glm::mat4(1.0f), glm::vec3(15.0f, movFutaba_y, 85.0f));
1319 model = glm::scale(model, glm::vec3(escalaFutaba1));
1320 animShader.setMat4("model", model);
1321 Futaba1.Draw(animShader);
1322
1323 // Dibujo Futaba 2
1324 model = glm::translate(glm::mat4(1.0f), glm::vec3(15.0f, -0.5f, 85.0f));
1325 model = glm::scale(model, glm::vec3(escalaFutaba2));
1326 animShader.setMat4("model", model);
1327 Futaba2.Draw(animShader);
```

```
1329 // Objetos estáticos
1330 staticShader.use();
1331 staticShader.setMat4("projection", projection);
1332 staticShader.setMat4("view", view);
1333
1334 // Piso
1335 model = glm::mat4(1.0f);
1336 model = glm::translate(model, glm::vec3(0.0f, -1.0f, 0.0f));
1337 model = glm::scale(model, glm::vec3(0.05f));
1338 staticShader.setMat4("model", model);
1339 piso.Draw(staticShader);
1340
1341 //Q-Front
1342 model = glm::mat4(1.0f);
1343 model = glm::translate(model, glm::vec3(-92.0f, -1.0f, 40.0f));
1344 model = glm::rotate(model, glm::radians(138.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1345 model = glm::scale(model, glm::vec3(2.5f, 2.5f, 3.1f));
1346 staticShader.setMat4("model", model);
1347 qfront.Draw(staticShader);
1348
1349 //Magnet
1350 model = glm::mat4(1.0f);
1351 model = glm::translate(model, glm::vec3(-65.0f, -0.7f, -38.0f));
1352 model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1353 model = glm::scale(model, glm::vec3(2.7f));
1354 staticShader.setMat4("model", model);
1355 magnet.Draw(staticShader);
```

```
1357 //Faro iluminación
1358 model = glm::mat4(1.0f);
1359 model = glm::translate(model, glm::vec3(-30.0f, -0.8f, 0.0f));
1360 model = glm::rotate(model, glm::radians(45.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1361 model = glm::scale(model, glm::vec3(2.3f));
1362 staticShader.setMat4("model", model);
1363 faro.Draw(staticShader);
1364
1365 //Torniquetes
1366 model = glm::mat4(1.0f);
1367 model = glm::translate(model, glm::vec3(55.0f, 2.3f, -2.0f));
1368 model = glm::scale(model, glm::vec3(2.35f));
1369 staticShader.setMat4("model", model);
1370 torniquetes.Draw(staticShader);
1371
1372 //Estacion
1373 model = glm::mat4(1.0f);
1374 model = glm::translate(model, glm::vec3(55.0f, -1.0f, -37.0f));
1375 model = glm::scale(model, glm::vec3(2.35f));
1376 staticShader.setMat4("model", model);
1377 estacion.Draw(staticShader);
1378
1379 //Espera
1380 model = glm::mat4(1.0f);
1381 model = glm::translate(model, glm::vec3(55.0f, -1.0f, -72.0f));
1382 model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1383 model = glm::scale(model, glm::vec3(2.35f));
1384 staticShader.setMat4("model", model);
1385 espera.Draw(staticShader);
```

```

1387     //Pirámide
1388     model = glm::mat4(1.0f);
1389     model = glm::translate(model, glm::vec3(25.0f, -0.7f, 60.0f));
1390     model = glm::scale(model, glm::vec3(3.5f));
1391     staticShader.setMat4("model", model);
1392     piramide.Draw(staticShader);

1393
1394     //Hachiko
1395     model = glm::mat4(1.0f);
1396     model = glm::translate(model, glm::vec3(20.0f, -0.9f, 0.0f));
1397     model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1398     model = glm::scale(model, glm::vec3(1.2f));
1399     staticShader.setMat4("model", model);
1400     hachiko.Draw(staticShader);

1401
1402     // Círculos cerca de Hachiko
1403     model = glm::mat4(1.0f);
1404     model = glm::translate(model, glm::vec3(15.0f, -0.9f, -5.0f));
1405     model = glm::rotate(model, glm::radians(70.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1406     model = glm::scale(model, glm::vec3(1.2f));
1407     staticShader.setMat4("model", model);
1408     circulo.Draw(staticShader);

1409
1410     model = glm::mat4(1.0f);
1411     model = glm::translate(model, glm::vec3(25.0f, -0.9f, -5.0f));
1412     model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1413     model = glm::scale(model, glm::vec3(1.2f));
1414     staticShader.setMat4("model", model);
1415     circulo.Draw(staticShader);

```

```

1438     //Cabina
1439     model = glm::mat4(1.0f);
1440     model = glm::translate(model, glm::vec3(movCabina_x, movCabina_y, movCabina_z));
1441     model = glm::rotate(model, glm::radians(90.0f + orientaCabina), glm::vec3(0.0f, 1.0f, 0.0f));
1442     model = glm::scale(model, glm::vec3(2.0f));
1443     staticShader.setMat4("model", model);
1444     cabina.Draw(staticShader);

1445
1446     //Vagon
1447     model = glm::mat4(1.0f);
1448     model = glm::translate(model, glm::vec3(movVagon_x, movVagon_y, movVagon_z));
1449     model = glm::rotate(model, glm::radians(90.0f + orientaVagon), glm::vec3(0.0f, 1.0f, 0.0f));
1450     model = glm::scale(model, glm::vec3(2.0f));
1451     staticShader.setMat4("model", model);
1452     vagon.Draw(staticShader);

1453
1454     //Reloj
1455     //Manecilla minutos
1456     model = glm::mat4(1.0f);
1457     model = glm::translate(model, glm::vec3(55.0f, 33.7f, -59.4f));
1458     model = glm::rotate(model, glm::radians(-giroMins), glm::vec3(0.0f, 0.0f, 1.0f));
1459     model = glm::scale(model, glm::vec3(2.35f));
1460     staticShader.setMat4("model", model);
1461     minutos.Draw(staticShader);
1462     //Manecilla horas
1463     model = glm::mat4(1.0f);
1464     model = glm::translate(model, glm::vec3(55.0f, 33.7f, -59.2f));
1465     model = glm::rotate(model, glm::radians(-giroHoras), glm::vec3(0.0f, 0.0f, 1.0f));
1466     model = glm::scale(model, glm::vec3(2.35f, 1.55f, 2.35f));
1467     staticShader.setMat4("model", model);
1468     horas.Draw(staticShader);

```

```
1470 //Ovni
1471 model = glm::mat4(1.0f);
1472 model = glm::translate(model, glm::vec3(movOvni_x, movOvni_y, movOvni_z));
1473 model = glm::rotate(model, glm::radians(orientaOvni), glm::vec3(0.0f, 1.0f, 0.0f));
1474 staticShader.setMat4("model", model);
1475 //staticShader.setVec3("dirLight.specular", glm::vec3(1.0f, 1.0f, 1.0f));
1476 ovni.Draw(staticShader);
1477
1478 //Plantitas, árboles y arbustos
1479 model = glm::mat4(1.0f);
1480 model = glm::translate(model, glm::vec3(65.0f, -0.7f, 95.0f));
1481 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1482 model = glm::scale(model, glm::vec3(1.2f));
1483 staticShader.setMat4("model", model);
1484 arbol.Draw(staticShader);
```

```
1514 model = glm::mat4(1.0f);
1515 model = glm::translate(model, glm::vec3(80.0f, -0.7f, 85.0f));
1516 model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1517 model = glm::scale(model, glm::vec3(2.2f));
1518 staticShader.setMat4("model", model);
1519 arbusto.Draw(staticShader);
1520
1521 model = glm::mat4(1.0f);
1522 model = glm::translate(model, glm::vec3(80.0f, -0.7f, 95.0f));
1523 model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1524 model = glm::scale(model, glm::vec3(2.4f));
1525 staticShader.setMat4("model", model);
1526 planta.Draw(staticShader);
```

```
1860 //Globos de dialogo
1861 //Globo Akechi
1862 model = glm::mat4(1.0f);
1863 model = glm::translate(model, glm::vec3(20.0f, 3.0f + mov_globoY, -30.0f + mov_globoXZ));
1864 model = glm::rotate(model, glm::radians(-90.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1865 model = glm::scale(model, glm::vec3(eglobo_Akechi));
1866 staticShader.setMat4("model", model);
1867 globo.Draw(staticShader);
1868 //Globo Joker
1869 model = glm::mat4(1.0f);
1870 model = glm::translate(model, glm::vec3(15.0 + mov_globoXZ, 3.0 + mov_globoY, -29.8f));
1871 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1872 model = glm::scale(model, glm::vec3(eglobo_Joker));
1873 staticShader.setMat4("model", model);
1874 globo.Draw(staticShader);
1875 //Globo Ann
1876 model = glm::mat4(1.0f);
1877 model = glm::translate(model, glm::vec3(15.0f + mov_globoXZ, 3.0f + mov_globoY, -29.0f));
1878 model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1879 model = glm::scale(model, glm::vec3(eglobo_Ann));
1880 staticShader.setMat4("model", model);
1881 globo.Draw(staticShader);
1882 //Globo Morgana
1883 model = glm::mat4(1.0f);
1884 model = glm::translate(model, glm::vec3(13.0f, 2.0f + mov_globoY, -30.0f + mov_globoXZ));
1885 model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1886 model = glm::scale(model, glm::vec3(eglobo_Morgana));
1887 staticShader.setMat4("model", model);
1888 globo.Draw(staticShader);
```

```

1890    //Vias del tren
1891
1892    model = glm::mat4(1.0f);
1893    model = glm::translate(model, glm::vec3(-114.5f, -0.5f, -72.0f));
1894    //model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1895    model = glm::scale(model, glm::vec3(3.5f));
1896    staticShader.setMat4("model", model);
1897    via.Draw(staticShader);

```

Para los modelos con jerarquía, como el camión, vemos código como el siguiente, que se basa en tomar un modelo como principal (para el caso del camión es la carrocería), se hace una matriz temporal sobre ese modelo y esa matriz temporal se utiliza para el dibujo de las demás partes (las ruedas, en el caso del camión).

**For models with hierarchy, such as the truck, we see code like the following, which is based on taking a model as the main model (in the case of the truck, it is the car body), a temporary matrix is made on that model and that temporary matrix is used for the drawing of the other parts (the wheels, in the case of the truck). This sets up the hierarchy.**

```

1606    // Camión gato
1607    model = glm::mat4(1.0f);
1608    model = glm::translate(model, glm::vec3(movCamion_x, movCamion_y, movCamion_z));
1609    tmp = model = glm::rotate(model, glm::radians(orientaCamion), glm::vec3(0.0f, 1.0f, 0.0f));
1610    staticShader.setMat4("model", model);
1611    camion.Draw(staticShader);
1612
1613    model = glm::translate(tmp, glm::vec3(1.7f, 0.0f, 2.0f));
1614    model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1615    model = glm::rotate(model, glm::radians(giroLlanta), glm::vec3(0.0f, 0.0f, 1.0f));
1616    staticShader.setMat4("model", model);
1617    rueda.Draw(staticShader); //delantera der
1618
1619    model = glm::translate(tmp, glm::vec3(-1.7f, 0.0f, 2.0f));
1620    model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1621    model = glm::rotate(model, glm::radians(giroLlanta), glm::vec3(0.0f, 0.0f, 1.0f));
1622    model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1623    staticShader.setMat4("model", model);
1624    rueda.Draw(staticShader); //delantera izq
1625
1626    model = glm::translate(tmp, glm::vec3(1.7f, 0.0f, -2.4f));
1627    model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1628    model = glm::rotate(model, glm::radians(giroLlanta), glm::vec3(0.0f, 0.0f, 1.0f));
1629    staticShader.setMat4("model", model);
1630    rueda.Draw(staticShader); //trasera der
1631
1632    model = glm::translate(tmp, glm::vec3(-1.7f, 0.0f, -2.4f));
1633    model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1634    model = glm::rotate(model, glm::radians(giroLlanta), glm::vec3(0.0f, 0.0f, 1.0f));
1635    model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1636    staticShader.setMat4("model", model);
1637    rueda.Draw(staticShader); //trasera izq

```

Para Morgana tenemos el siguiente código.

**For Morgana we have the following code.**

```

1639     // Morgana
1640     //Torso
1641     model = glm::translate(glm::mat4(1.0f), glm::vec3(0.0f, 0.0f, 0.0f));
1642     model = glm::translate(model, glm::vec3(13.0f, 0.0f, -30.0f));
1643     tmp = model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 1.0f, 0.0));
1644     model = glm::scale(model, glm::vec3(0.3f));
1645     staticShader.setMat4("model", model);
1646     torsoMorgana.Draw(staticShader);
1647     //Brazo derecho
1648     model = glm::translate(tmp, glm::vec3(-0.2f, 0.2f, 0.0f));
1649     //model = glm::translate(model, glm::vec3(0.75f, 2.5f, 0));
1650     model = glm::rotate(model, glm::radians(45.0f), glm::vec3(0.0f, 0.0f, 1.0f));
1651     model = glm::scale(model, glm::vec3(0.3f));
1652     staticShader.setMat4("model", model);
1653     brazoMorgana.Draw(staticShader);
1654     //Brazo izquierdo
1655     model = glm::translate(tmp, glm::vec3(0.2f, 0.2f, 0.0f));
1656     //model = glm::translate(model, glm::vec3(0.75f, 2.5f, 0));
1657     model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1658     model = glm::rotate(model, glm::radians(45.0f), glm::vec3(0.0f, 0.0f, 1.0f));
1659     model = glm::scale(model, glm::vec3(0.3f));
1660     staticShader.setMat4("model", model);
1661     brazoMorgana.Draw(staticShader);
1662     //Cabeza
1663     model = glm::translate(tmp, glm::vec3(0.0f, 0.35f, -0.05f));
1664     model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0));
1665     model = glm::translate(model, glm::vec3(0.0f, 0.0f, 0));
1666     model = glm::scale(model, glm::vec3(0.3f));
1667     staticShader.setMat4("model", model);
1668     cabezaMorgana.Draw(staticShader);

1669     //Pierna Izq
1670     model = glm::translate(tmp, glm::vec3(0.15f, -0.35f, 0.0f));
1671     model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0));
1672     model = glm::rotate(model, glm::radians(0.0f), glm::vec3(1.0f, 0.0f, 0.0f));
1673     model = glm::scale(model, glm::vec3(0.3f));
1674     staticShader.setMat4("model", model);
1675     piernaMorgana.Draw(staticShader);
1676     //Pierna Der
1677     model = glm::translate(tmp, glm::vec3(-0.15f, -0.35f, 0.0f));
1678     model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0));
1679     model = glm::rotate(model, glm::radians(0.0f), glm::vec3(1.0f, 0.0f, 0.0f));
1680     model = glm::scale(model, glm::vec3(0.3f));
1681     staticShader.setMat4("model", model);
1682     piernaMorgana.Draw(staticShader);

```

Para los demás personajes, tenemos una estructura bastante similar:

**For the other characters, we have a fairly similar code structure:**

```

1726 //Joker
1727 //Torso
1728 model = glm::translate(glm::mat4(1.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1729 model = glm::translate(model, glm::vec3(15.0f, 0.0f, -29.8f));
1730 tmp = model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1731 model = glm::scale(model, glm::vec3(0.5f));
1732 staticShader.setMat4("model", model);
1733 torsoJoker.Draw(staticShader);
1734 //Brazo derecho
1735 model = glm::translate(tmp, glm::vec3(-0.51f, 0.5f, 0.0f));
1736 model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
1737 model = glm::scale(model, glm::vec3(0.5f));
1738 staticShader.setMat4("model", model);
1739 brazoJoker.Draw(staticShader);
1740 //Brazo izquierdo
1741 model = glm::translate(tmp, glm::vec3(0.51f, 0.5f, 0.0f));
1742 model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1743 model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
1744 model = glm::scale(model, glm::vec3(0.5f));
1745 staticShader.setMat4("model", model);
1746 brazoJoker.Draw(staticShader);
1747 //Cabeza
1748 model = glm::translate(tmp, glm::vec3(0.0f, 0.70f, 0.0f));
1749 model = glm::rotate(model, glm::radians(giroCabezaJoker_y), glm::vec3(0.0f, 1.0f, 0.0f));
1750 model = glm::translate(model, glm::vec3(0.0f, 0.0f, 0));
1751 model = glm::scale(model, glm::vec3(0.5f));
1752 staticShader.setMat4("model", model);
1753 cabezaJoker.Draw(staticShader);

1754 //Pierna Der
1755 model = glm::translate(tmp, glm::vec3(-0.28f, -0.69f, 0.0f));
1756 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1757 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(1.0f, 0.0f, 0.0f));
1758 model = glm::scale(model, glm::vec3(0.5f));
1759 staticShader.setMat4("model", model);
1760 piernaJoker.Draw(staticShader);
1761 //Pierna Izq
1762 model = glm::translate(tmp, glm::vec3(0.28f, -0.69f, 0.0f));
1763 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1764 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(1.0f, 0.0f, 0.0f));
1765 model = glm::scale(model, glm::vec3(0.5f));
1766 staticShader.setMat4("model", model);
1767 piernaJoker.Draw(staticShader);

```

```

1769 // Akechi
1770 //Torso
1771 model = glm::translate(glm::mat4(1.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1772 model = glm::translate(model, glm::vec3(20.0f, 0.0f, -30.0f));
1773 tmp = model = glm::rotate(model, glm::radians(-90.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1774 model = glm::scale(model, glm::vec3(0.5f));
1775 staticShader.setMat4("model", model);
1776 torsoAkechi.Draw(staticShader);
1777 //Brazo derecho
1778 model = glm::translate(tmp, glm::vec3(-0.51f, 0.5f, 0.0f));
1779 model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
1780 model = glm::rotate(model, glm::radians(giroBrazoAkechi_x), glm::vec3(0.0f, 1.0f, 0.0f));
1781 model = glm::scale(model, glm::vec3(0.5f));
1782 staticShader.setMat4("model", model);
1783 brazoAkechi.Draw(staticShader);
1784 //Brazo izquierdo
1785 model = glm::translate(tmp, glm::vec3(0.51f, 0.5f, 0.0f));
1786 model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1787 model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
1788 model = glm::rotate(model, glm::radians(-giroBrazoAkechi_x), glm::vec3(0.0f, 1.0f, 0.0f));
1789 model = glm::scale(model, glm::vec3(0.5f));
1790 staticShader.setMat4("model", model);
1791 brazoAkechi.Draw(staticShader);
1792 //Cabeza
1793 model = glm::translate(tmp, glm::vec3(0.0f, 0.71f, 0.0f));
1794 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1795 model = glm::translate(model, glm::vec3(0.0f, 0.0f, 0.0f));
1796 model = glm::scale(model, glm::vec3(0.5f));
1797 staticShader.setMat4("model", model);
1798 cabezaAkechi.Draw(staticShader);

1799 //Pierna Der
1800 model = glm::translate(tmp, glm::vec3(-0.28f, -0.69f, 0.0f));
1801 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1802 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(1.0f, 0.0f, 0.0f));
1803 model = glm::scale(model, glm::vec3(0.5f));
1804 staticShader.setMat4("model", model);
1805 piernaAkechi.Draw(staticShader);
1806 //Pierna Izq
1807 model = glm::translate(tmp, glm::vec3(0.28f, -0.69f, 0.0f));
1808 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1809 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(1.0f, 0.0f, 0.0f));
1810 model = glm::scale(model, glm::vec3(0.5f));
1811 staticShader.setMat4("model", model);
1812 piernaAkechi.Draw(staticShader);

```

```

1814 // Ann
1815 //Torso
1816 model = glm::translate(glm::mat4(1.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1817 model = glm::translate(model, glm::vec3(15.0f, 0.0f, -29.0f));
1818 tmp = model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1819 model = glm::scale(model, glm::vec3(0.5f));
1820 staticShader.setMat4("model", model);
1821 torsoAnn.Draw(staticShader);
1822 //Brazo derecho
1823 model = glm::translate(tmp, glm::vec3(-0.51f, 0.5f, 0.0f));
1824 model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1825 model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
1826 model = glm::rotate(model, glm::radians(giroBrazoDerechoAnn_x), glm::vec3(1.0f, 0.0f, 0.0f));
1827 model = glm::rotate(model, glm::radians(giroBrazoDerechoAnn_y), glm::vec3(0.0f, 1.0f, 0.0f));
1828 model = glm::scale(model, glm::vec3(0.5f));
1829 staticShader.setMat4("model", model);
1830 brazoAnn.Draw(staticShader);
1831 //Brazo izquierdo
1832 model = glm::translate(tmp, glm::vec3(0.51f, 0.5f, 0.0f));
1833 model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1834 model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
1835 model = glm::scale(model, glm::vec3(0.5f));
1836 staticShader.setMat4("model", model);
1837 brazoAnn.Draw(staticShader);
1838 //Cabeza
1839 model = glm::translate(tmp, glm::vec3(0.0f, 0.70f, 0.0f));
1840 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1841 model = glm::translate(model, glm::vec3(0.0f, 0.0f, 0));
1842 model = glm::scale(model, glm::vec3(0.5f));
1843 staticShader.setMat4("model", model);

1845 //Pierna Izq
1846 model = glm::translate(tmp, glm::vec3(0.27f, -0.71f, 0.02f));
1847 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1848 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(1.0f, 0.0f, 0.0f));
1849 model = glm::scale(model, glm::vec3(0.5f));
1850 staticShader.setMat4("model", model);
1851 pierna2Ann.Draw(staticShader);
1852 //Pierna Der
1853 model = glm::translate(tmp, glm::vec3(-0.27f, -0.71f, 0.07f));
1854 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(0.0f, 1.0f, 0.0f));
1855 model = glm::rotate(model, glm::radians(0.0f), glm::vec3(1.0f, 0.0f, 0.0f));
1856 model = glm::scale(model, glm::vec3(0.5f));
1857 staticShader.setMat4("model", model);
1858 piernaAnn.Draw(staticShader);

```

Básicamente, este es el método de dibujo de todos los modelos. Algunos tienen ligeras variaciones porque son animados (como Futaba). Y también ocurre que se pueden apreciar variables en las operaciones geométricas de los mismos, pues estas variables nos permitirán crear animaciones, como se describe más adelante.

**Basically, this is the drawing method for all models. Some have slight variations in the shaders because they are animated (like Futaba). Also, some variables can be seen in geometric operations, since these variables will allow us to create animations, as described later.**



Vista aérea de los modelos.

**Aerial view of the models.**

## Recorrido / Traveling

Para este apartado se toman en cuenta las cámaras implementadas (especialmente una cámara ligada al piso). Contamos con 3 cámaras:

- Cámara principal, que cuenta con movimiento libre en todos los ejes. Esto permite recorrer todo el escenario desde diferentes puntos de vista.
- Cámara ligada al plano XZ, o cámara de piso. Esta cámara cuenta con una elevación muy leve, muy cercana al piso, pero no permite avanzar verticalmente por el escenario.
- Cámara aérea. En automático avanza al centro del escenario, pero con una gran altura y observando hacia abajo. Esto permite observar todos los elementos del escenario.

La implementación de estas cámaras viene de la biblioteca camera.h, hallada en la carpeta include del proyecto.

**For this section, the implemented cameras are taken into account (especially a camera linked to the floor). We have 3 cameras:**

- Main camera, which has free movement in all axes. This allows you to move freely around the entire stage from different points of view.
- Camera linked to the XZ plane, or floor camera. This camera has a very slight elevation, near the floor, but does not allow you to move vertically across the stage.
- Aerial camera. Automatically, advances to the center of the stage, but with a great height and looking down. This allows you to observe all the elements of the stage.

**The implementation of these cameras comes from the camera.h library, found in the include folder of the project.**

Nombre	Fecha de modificación	Tipo	Tamaño
assimp	10/04/2022 05:31 p.m.	Carpeta de archivos	
glad	10/04/2022 05:31 p.m.	Carpeta de archivos	
GLFW_no	10/04/2022 05:31 p.m.	Carpeta de archivos	
glm	23/05/2022 11:00 a.m.	Carpeta de archivos	
irrKlang	06/05/2022 04:22 p.m.	Carpeta de archivos	
KHR	10/04/2022 05:31 p.m.	Carpeta de archivos	
SDL	10/04/2022 05:31 p.m.	Carpeta de archivos	
camera.h	21/05/2022 09:10 p.m.	Archivo de origen ...	5 KB

Se incluye al inicio del proyecto.

**It is included at the beginning of the project.**

26      ||#include <camera.h>

Con esto, creamos nuestro objeto cámara y nuestras variables para manejar la posición y ángulos de la cámara.

**With this, we create our camera object and our variables to handle the position and angles of the camera.**

```

52     // Cámara
53     float auxx = 0.0f, auxy = 10.0f, auxz = 100.0f, auxpitch=0.0f;
54     Camera camera(glm::vec3(auxx, auxy, auxz)); //Cámera libre
55     float MovementSpeed = 5.0f;
56     float lastX = SCR_WIDTH / 2.0f;
57     float lastY = SCR_HEIGHT / 2.0f;
58     bool firstMouse = true;
59     bool camaraPiso = false;
60     bool camaraAerea = false;
61     bool camaraLibre2 = false;
62     bool camaraLibre1 = false;

```

Igualmente, tenemos nuestras funciones relacionadas al mouse para poder controlar la dirección de la vista y así también modificar la dirección del movimiento de la cámara.

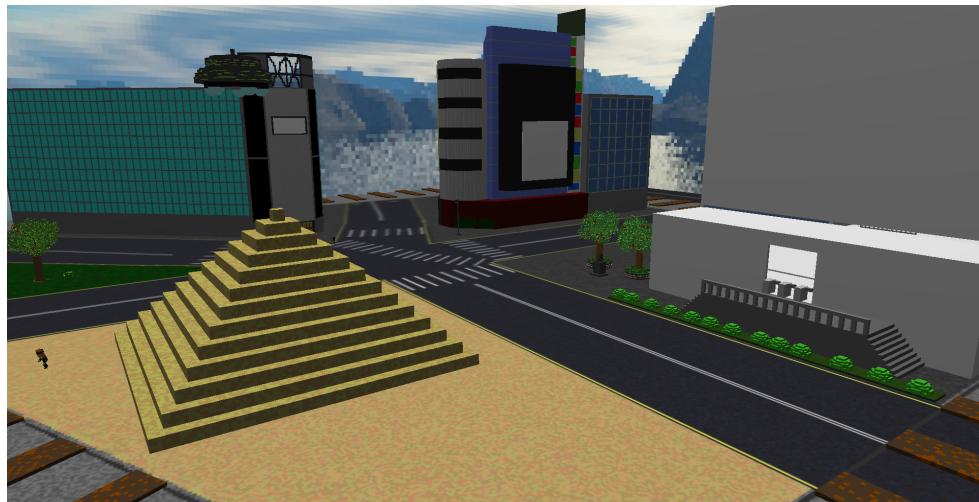
**Likewise, we have our functions related to the mouse to be able to control the direction of the view and thus also modify the camera movement direction.**

```

2546 void mouse_callback(GLFWwindow* window, double xpos, double ypos)
2547 {
2548     if (firstMouse)
2549     {
2550         lastX = xpos;
2551         lastY = ypos;
2552         firstMouse = false;
2553     }
2554
2555     float xoffset = xpos - lastX;
2556     float yoffset = lastY - ypos; // reversed since y-coordinates go from bottom to top
2557
2558     lastX = xpos;
2559     lastY = ypos;
2560
2561     camera.ProcessMouseMovement(xoffset, yoffset);
2562 }
2563 // glfw: whenever the mouse scroll wheel scrolls, this callback is called
2564 // -----
2565 void scroll_callback(GLFWwindow* window, double xoffset, double yoffset)
2566 {
2567     camera.ProcessMouseScroll(yoffset);
2568 }
```

Así tenemos nuestra cámara libre.

**And so we have our main camera.**



Muestra de cámara libre

#### Free camera sample

Para la cámara ligada al piso y aérea, asignamos las teclas C y K, respectivamente. Al activarse alguno de estos booleanos, la cámara pasará a tener el comportamiento descrito antes.

**For the floor camera and aerial camera, we assign the keys C and K, respectively. When one of these booleans is activated, the camera will have the behavior described above.**

```

2396     //Para activar cámara en xz
2397     if (glfwGetKey(window, GLFW_KEY_C) == GLFW_PRESS)
2398         camaraPiso = !camaraPiso;
2399
2400     //Para activar cámara aerea
2401     if (glfwGetKey(window, GLFW_KEY_K) == GLFW_PRESS)
2402         camaraAerea = !camaraAerea;

```

La manera en que estos booleanos afectan es en el bucle de renderización del proyecto.

Si activamos la cámara de piso, entonces la posición en el eje Y siempre se mantendrá en una altura muy baja, pero es posible moverse en X o Z. En cambio, para la cámara aérea, se mantiene en una posición muy alta en Y, sin posibilidad de moverse en X, Y o Z, y siempre mirando hacia abajo.

**The way these booleans affect the behavior is in the render loop of the project.**

**If we activate the floor camera, then the position on the Y axis will always stay at a very low height, but it is possible to move in X or Z. Instead, for the aerial camera, it stays at a very high position in Y, without the possibility of moving in X, Y or Z, and always looking down.**

```

1180     //Cámara en xz:
1181     if (camaraPiso) {
1182         camera.Position.y = 1.0f;
1183         camaraLibre1 = true;
1184     }
1185     else {
1186         if (camaraLibre1) {
1187             camera.Position.y = auxy;
1188             camaraLibre1 = false;
1189         }
1190     }
1191
1192
1193     if (camaraAerea) {
1194         camera.Position.y = 325.0f;
1195         camera.Position.x = 0.0f;
1196         camera.Position.z = 0.0f;
1197         camera.Pitch=-90.0f;
1198         camera.ProcessMouseMovement(0, 0);
1199         camaraLibre2 = true;
1200     }
1201     else {
1202         if (camaraLibre2) {
1203             Camera camera2(glm::vec3(auxx, auxy, auxz));
1204             camera = camera2;
1205             camaraLibre2 = false;
1206         }
1207     }

```

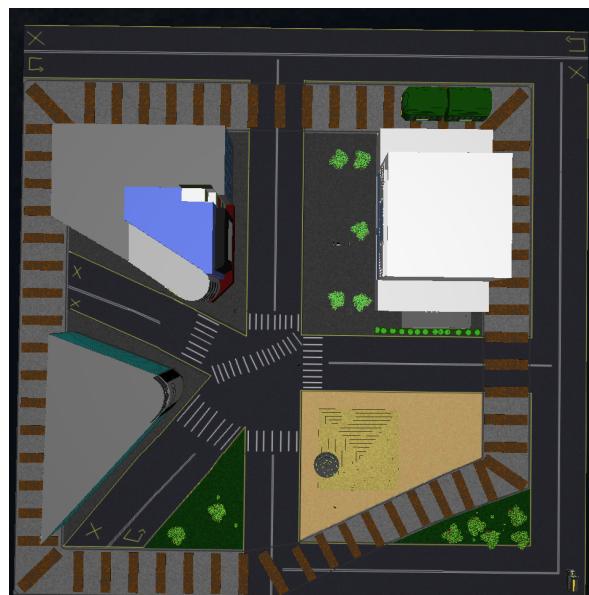
Las cámaras de piso y aérea se verán así:

The ground and aerial cameras will look like this:



Muestra de cámara ligada al piso

Floor camera sample



Muestra de cámara aérea

Aerial camera sample

## Illuminación / Illumination

Para la iluminación hacemos uso del staticShader. Lo primero es usar ese shader y después ponemos nuestra luz direccional (tenemos 1 luz direccional, 2 de tipo spotlight y 3 luces puntuales).

Para la luz direccional ponemos a nuestra cámara como punto de referencia e indicamos que queremos que la dirección sea hacia abajo en Y y negativo en Z (es decir, que la luz vaya en dirección hacia la cámara). Posteriormente, modificamos sus componentes ambiental, difusa y especular.

**For the illumination we make use of the staticShader. The first thing to do is to use that shader and then we put our directional light (we have 1 directional light, 2 spotlight type and 3 punctual lights).**

**For the directional light we set our camera as the reference point and indicate that we want the direction to be downward in Y and negative in Z (that is, that the light goes in the direction of the camera). Subsequently, we modify its ambient, diffuse and specular components.**

```
71      // Iluminación
72      glm::vec3 lightPosition(0.0f, 4.0f, -10.0f);
73      glm::vec3 lightDirection(0.0f, -1.0f, -1.0f);
74      glm::vec3 lightPositionSun(0.0f, 550.0f, 0.0f);
75      glm::vec3 luzColor(0.0f, 0.0f, 0.0f);
```

```
1221      // don't forget to enable shader before setting uniforms
1222      staticShader.use();
1223      //Setup Advanced Lights
1224      // Iluminación
1225      staticShader.setVec3("viewPos", camera.Position);
1226      staticShader.setVec3("dirLight.direction", lightDirection);
1227      staticShader.setVec3("dirLight.ambient", glm::vec3(0.125f, 0.125f, 0.125f));
1228      staticShader.setVec3("dirLight.diffuse", glm::vec3(0.125f, 0.125f, 0.125f));
1229      staticShader.setVec3("dirLight.specular", glm::vec3(0.0f, 0.0f, 0.0f));
```

Para las luces puntuales, tenemos una que simula ser el sol y otras 2 que van enfrente del camión (en sus faros delanteros). El procedimiento es el mismo para todas: Les damos una posición en el entorno, modificamos sus componentes ambiente, difusa y especular, así como modificamos su valores constante, lineal y cuadrático, para que abarquen una mayor distancia o mayor intensidad en la iluminación.

**For the punctual lights, we have one that simulates the sun and another 2 that go in front of the truck (in its headlights). The procedure is the same for all of them: We give them a position in the environment, we modify their ambient, diffuse and specular components, as well as we modify their constant, linear and quadratic values, so that they cover a greater distance or greater intensity in illumination.**

```

1231 //Sol
1232 staticShader.setVec3("pointLight[0].position", lightPositionSun);
1233 staticShader.setVec3("pointLight[0].ambient", glm::vec3(0.0f, 0.0f, 0.0f));
1234 staticShader.setVec3("pointLight[0].diffuse", glm::vec3(1.0f, 1.0f, 1.0f));
1235 staticShader.setVec3("pointLight[0].specular", glm::vec3(0.3f, 0.3f, 0.3f));
1236 staticShader.setFloat("pointLight[0].constant", 0.08f);
1237 staticShader.setFloat("pointLight[0].linear", 0.0009f);
1238 staticShader.setFloat("pointLight[0].quadratic", 0.000004f);

1240 //Luz para el gato camion 1
1241 //staticShader.setVec3("pointLight[1].position", glm::vec3(movCamion_x - 3, movCamion_y + 3, movCamion_z - 6));
1242 staticShader.setVec3("pointLight[1].position", glm::vec3(movCamion_x + movCamionLuz_x, movCamion_y + movCamionLuz_y, movCamion_z + movCamionLuz_z));
1243 staticShader.setVec3("pointLight[1].ambient", glm::vec3(ilumCamionR, ilumCamionG, ilumCamionB));
1244 staticShader.setVec3("pointLight[1].diffuse", glm::vec3(ilumCamionR, ilumCamionG, ilumCamionB));
1245 staticShader.setVec3("pointLight[1].specular", glm::vec3(ilumCamionR, ilumCamionG, ilumCamionB));
1246 staticShader.setFloat("pointLight[1].constant", 0.08f);
1247 staticShader.setFloat("pointLight[1].linear", 0.009f);
1248 staticShader.setFloat("pointLight[1].quadratic", 0.5f);
1249

1250 //Luz para el gato camion 2
1251 //staticShader.setVec3("pointLight[2].position", glm::vec3(movCamion_x + 3, movCamion_y + 3, movCamion_z - 6));
1252 staticShader.setVec3("pointLight[2].position", glm::vec3(movCamion_x + movCamionLuz2_x, movCamion_y + movCamionLuz_y, movCamion_z + movCamionLuz_z));
1253 staticShader.setVec3("pointLight[2].ambient", glm::vec3(ilumCamionR, ilumCamionG, ilumCamionB));
1254 staticShader.setVec3("pointLight[2].diffuse", glm::vec3(ilumCamionR, ilumCamionG, ilumCamionB));
1255 staticShader.setVec3("pointLight[2].specular", glm::vec3(ilumCamionR, ilumCamionG, ilumCamionB));
1256 staticShader.setFloat("pointLight[2].constant", 0.08f);
1257 staticShader.setFloat("pointLight[2].linear", 0.009f);
1258 staticShader.setFloat("pointLight[2].quadratic", 0.5f);

```

Las luces puntuales se aprecian así en el camión, aunque es difícil apreciar la luz del Sol en una imagen:

**The punctual lights are appreciated like this in the truck, although it is difficult to appreciate the light of the Sun in an image:**



Para nuestras luces spotlight, el procedimiento es muy similar al de las luces puntuales, aunque con algunos parámetros extra, como el ángulo de inicio y el ángulo de terminación (cutoff y outer cutoff). Así como también les colocamos una dirección (en este caso, ambas spotlights apuntan hacia abajo).

**For our spotlights, the procedure is quite similar to that of punctual lights, but with some extra parameters, such as the cutoff and outer cutoff. Also, we assign a direction to them (in this case, the two spotlights point downwards).**

```

1260 //Luz de ovni
1261 staticShader.setVec3("spotLight[0].position", glm::vec3(movOvni_x, movOvni_y+5.0f, movOvni_z));
1262 staticShader.setVec3("spotLight[0].direction", glm::vec3(0.0f, -1.0f, 0.0f));
1263 staticShader.setVec3("spotLight[0].ambient", glm::vec3(0.0f, ilumOvni, 0.0f));
1264 staticShader.setVec3("spotLight[0].diffuse", glm::vec3(0.0f, ilumOvni, 0.0f));
1265 staticShader.setVec3("spotLight[0].specular", glm::vec3(0.0f, ilumOvni, 0.0f));
1266 staticShader.setFloat("spotLight[0].cutOff", glm::cos(glm::radians(20.0f)));
1267 staticShader.setFloat("spotLight[0].outerCutOff", glm::cos(glm::radians(40.0f)));
1268 staticShader.setFloat("spotLight[0].constant", 1.0f);
1269 staticShader.setFloat("spotLight[0].linear", 0.009f);
1270 staticShader.setFloat("spotLight[0].quadratic", 0.005f);
1271
1272 //Luz del faro
1273 staticShader.setVec3("spotLight[1].position", glm::vec3(-29.0f, 9.0f, 0.5f));
1274 staticShader.setVec3("spotLight[1].direction", glm::vec3(0.0f, -1.0f, -0.0f));
1275 staticShader.setVec3("spotLight[1].ambient", glm::vec3(0.0f, 0.0f, 0.0f));
1276 staticShader.setVec3("spotLight[1].diffuse", glm::vec3(ilumFaro, ilumFaro, 0.0f));
1277 staticShader.setVec3("spotLight[1].specular", glm::vec3(ilumFaro, ilumFaro, 0.0f));
1278 staticShader.setFloat("spotLight[1].cutOff", glm::cos(glm::radians(20.0f)));
1279 staticShader.setFloat("spotLight[1].outerCutOff", glm::cos(glm::radians(60.0f)));
1280 staticShader.setFloat("spotLight[1].constant", 1.0f);
1281 staticShader.setFloat("spotLight[1].linear", 0.0009f);
1282 staticShader.setFloat("spotLight[1].quadratic", 0.005f);

```

Las luces spotlight se verán así:

**The spotlights will look like this:**



Como se vio en capturas anteriores, las luces tienen valores en variables, y las luces spotlight del camión están “apagadas” (en 0s). Estas luces se encenderán al activar animaciones y llegar a ciertos estados. Aunque la luz del faro se encenderá/apagará con la tecla F.

**As seen in previous screenshots, the lights have values inside some variables, and the truck's spotlights are "turned off" (values of 0). These lights will light up when triggering animations and reaching certain states. Though the beacon light will be turned on/off by pressing the F key.**

```
2399      //Activar luz
2400      if (key == GLFW_KEY_F && action == GLFW_PRESS) {
2401          faroOn ^= true;
2402          if (faroOn)
2403              ilumFaro = 1.0f;
2404          else
2405              ilumFaro = 0.0f;
2406 }
```

Para el camión, se encenderán las luces en blanco en la mayoría de estados:

**For the truck, the lights will turn on in most states:**

```
437      // Para gato camion (Animación 3)
438      if (animacion_camion) {
439          girollanta += 0.2f;
440          switch (estado_camion) {
441              case 1:
442                  if (movCamion_z >= 70.0f) {
443                      girollanta += 1.0f;
444                      movCamion_z -= 1.0f;
445                      movCamionLuz_z = -6.0f;
446                      movCamionLuz2_z = -6.0f;
447                      ilumCamionR = 1.0f;
448                      ilumCamionG = 1.0f;
449                      ilumCamionB = 1.0f;
450                  }
451                  else {
452                      estado_camion++;
453                  }
454          break;
```

Para el ovni, se encenderá el spotlight en verde cuando abduce y devuelve a Futaba.

**For the UFO, the spotlight will turn green when it abducts and returns Futaba.**

```

653     case 1:
654         //Desaparece Futaba 2 y aparece Futaba 1
655         escalaFutaba2 = 0.0f;
656         escalaFutaba1 = 0.35f;
657         estado_Ovni = 2;
658         ilumOvni = 1.0f; //Encendemos luz de ovni
659         break;
660     case 2:
661         //Futaba 1 viaja hacia arriba con escala, rotación y traslación en Y
662         if (escalaFutaba1 >= 0) {
663             //Movimiento y decremeneto
664             escalaFutaba1 -= 0.001;
665             movFutaba_y += 0.1;
666         }
667         else {
668             estado_Ovni = 3;
669         }
670         break;
671     case 3:
672         //Pequeño delay xD
673         if (contOvni <= 120) {
674             contOvni++;
675             ilumOvni = 0.0f; //Apagamos luz de ovni
676         }
677         else {
678             ilumOvni = 1.0f; //Encendemos luz de ovni
679             estado_Ovni = 4;
680             contOvni = 0;
681         }
682         break;

```

Y el sol se moverá cuando se active su animación:

**And the sun will move when its animation is activated:**

```

307     //Animación del sol (Animación 0)
308     if (animacion_sol) {
309         lightPositionSun.x = 500.0f * cos(movSol);
310         lightPositionSun.y = 500.0f * sin(movSol);
311         if (lightPositionSun.y < 0 || lightPositionSun.x < 0) {
312             movSol += 0.005f;
313         }
314         else {
315             movSol += 0.0025f;
316         }
317     }

```

## Animación / Animation

En este apartado, contamos con 8 animaciones, de las cuales 1 usa el método de keyframes, la mayoría básicas (basadas en transformaciones y bandera) y otras pocas son complejas. Para mayor información de qué se puede observar de ellas, leer el manual de usuario.

Comenzando por la animación 0, acorde a su tecla asignada, se trata del movimiento de la luz puntual que funge como sol. Al presionar la tecla 0, se activa el booleano que permite la ejecución de la animación en nuestra función de animaciones (animate) que se ejecuta en el bucle de ejecución.

In this section, we have 8 animations, of which 1 uses the keyframe method, most of these animations are basic animations (based on transformations and flags) and a few others are complex or advanced. For more information on what can be observed from them, read the user manual.

Starting with animation 0, according to its assigned key, it is the movement of the point light that acts as the sun. Pressing the 0 key activates the boolean that allows the execution of the animation in our animations function (animate) that is executed in the execution loop.

```
2408     //Animación 0: Luz del sol
2409     if (key == GLFW_KEY_0 && action == GLFW_PRESS) {
2410         animacion_sol ^= true;
2411     }
```

1214 | | | animate();

Esta animación consiste en el sol recorriendo, por medio de una fórmula, una ruta circular en los ejes X e Y, a lo largo del escenario. Cuando el sol se encuentra arriba del escenario, entonces irá más lento, pero si va por debajo del escenario se moverá el doble de rápido.

This animation consists of the sun traveling, by means of a formula, in a circular path in the X and Y axes, along the stage. When the sun is above the stage then it will go slower, but if it goes below the stage it will move twice as fast.

```
305     void animate(void)
306     {
307         //Animación del sol (Animación 0)
308         if (animacion_sol) {
309             lightPositionSun.x = 500.0f * cos(movSol);
310             lightPositionSun.y = 500.0f * sin(movSol);
311             if (lightPositionSun.y < 0 || lightPositionSun.x < 0) {
312                 movSol += 0.005f;
313             }
314             else {
315                 movSol += 0.0025f;
316             }
317         }
```

Para la animación 1, tenemos unas manecillas del reloj en la estación de tren. Los minutos se mueven a una velocidad y las horas a 1/12 de esa velocidad. Se activa al presionar la tecla 1. Al presionar esta tecla también reiniciamos los valores de inicio.

**For animation 1, we have clock hands at the train station. The minutes hand moves at one speed and the hours hand moves at 1/12 of that speed. It is activated by pressing the key 1. Pressing this key also resets the start values.**

```
2413      //Animacion 1: Manecillas del reloj
2414      if (key == GLFW_KEY_1 && action == GLFW_PRESS) {
2415          animacion_reloj ^= true;
2416          giroHoras = 0;
2417          giroMins = 0;
2418      }
```

```
319      //Para reloj: (Animación 1)
320      if (animacion_reloj) {
321          giroMins += 0.3f;
322          giroHoras += 0.025f;
323      }
```

Para la animación 2, la activamos al presionar la tecla 2. Así mismo, ponemos los valores iniciales.

**For animation 2, we activate it by pressing the 2 key. Likewise, we put the initial values.**

```
2420      //Animación 2: Movimiento del tren
2421      if (key == GLFW_KEY_2 && action == GLFW_PRESS) {
2422          animacion_tren ^= true;
2423          estadoCabina = 0;
2424          estadoVagon = 0;
2425          orientaCabina = 0.0f;
2426          movCabina_x = 51.0f;
2427          movCabina_z = -90.0f;
2428          orientaVagon = 0.0f;
2429          movVagon_x = 70.0f;
2430          movVagon_y = 0.2f;
2431          movVagon_z = -90.0f;
2432      }
```

En esta animación veremos al tren moverse por todo el escenario. Consta de 5 estados, aunque cabe destacar que esto ocurre tanto para el vagón como para la cabina.

**In this animation we will see the train moving around the stage. It consists of 5 states, although it should be noted that this occurs for both the wagon and the cabin.**

```

325      //Para tren (Animación 2)
326      if (animacion_tren) {
327          switch (estadoCabina) {
328              case 0: //Estado inicial hacia izquierda
329                  orientaCabina = 0.0f;
330                  if (movCabina_x >= -115.0f) {
331                      movCabina_x -= 1.0f;
332                  }
333                  else {
334                      estadoCabina = 1;
335                      orientaCabina = 45.0f;
336                  }
337                  break;
338              case 1: //Hacia abajo
339                  orientaCabina = 90.0f;
340                  if (movCabina_z <= 115.0f) {
341                      movCabina_z += 1.0f;
342                  }
343                  else {
344                      estadoCabina = 2;
345                      orientaCabina = 135.0f;
346                  }
347                  break;
348              case 2: //Hacia la derecha
349                  orientaCabina = 180.0f;
350                  if (movCabina_x <= -15.0f) {
351                      movCabina_x += 1.0f;
352                  }
353                  else {

```

```

354                  estadoCabina = 3;
355                  orientaCabina = 192.615f;
356              }
357              break;
358          case 3: //Giro raro
359              orientaCabina = 205.23f;
360              if (movCabina_x <= 90.0f) {
361                  movCabina_x += 1.0f;
362                  movCabina_z -= 0.47116f;
363              }
364              else {
365                  movCabina_z = 65.0f;
366                  estadoCabina = 4;
367                  orientaCabina = 237.615;
368              }
369              break;
370          case 4: //Hacia arriba
371              orientaCabina = 270.0f;
372              if (movCabina_z >= -90.0f) {
373                  movCabina_z -= 1.0f;
374              }
375              else {
376                  estadoCabina = 0;
377                  orientaCabina = 305.0f;
378              }
379              break;
380          }
381          switch (estadoVagon) {
382              case 0: //Hacia la izquierda
383                  orientaVagon = 0.0f;
384                  if (movVagon_x >= -115.0f) {

```

385	movVagon_x -= 1.0f;
386	}
387	else {
388	estadoVagon = 1;
389	orientaVagon = 45.0f;
390	}
391	break;
392	case 1: //Hacia abajo
393	orientaVagon = 90.0f;
394	if (movVagon_z <= 115.0f) {
395	movVagon_z += 1.0f;
396	}
397	else {
398	estadoVagon = 2;
399	orientaVagon = 135.0f;
400	}
401	break;
402	case 2: //Hacia la derecha
403	orientaVagon = 180.0f;
404	if (movVagon_x <= -15.0f) {
405	movVagon_x += 1.0f;
406	}
407	else {
408	estadoVagon = 3;
409	orientaVagon = 192.615f;
410	}
411	break;
412	case 3: //Giro raro
413	orientaVagon = 205.23f;
414	if (movVagon_x <= 90.0f) {
415	movVagon_x += 1.0f;

```

416           movVagon_z -= 0.47116f;
417       }
418   } else {
419     movVagon_z = 65.0f;
420     estadoVagon = 4;
421     orientaVagon = 237.615;
422   }
423   break;
424 case 4: //Hacia arriba
425   orientaVagon = 270.0f;
426   if (movVagon_z >= -90.0f) {
427     movVagon_z -= 1.0f;
428   }
429   else {
430     estadoVagon = 0;
431     orientaVagon = 305.0f;
432   }
433   break;
434 }
435 }
```

Para la animación 3, la activamos al presionar la tecla 3. Al igual que con las anteriores, tenemos un reinicio de las variables cada vez que presionamos esta tecla.

**For animation 3, we activate it by pressing the key 3. As with the previous ones, we have a reset of the variables each time we press this key.**

```

2434 //Animación 3: Camión
2435 if (key == GLFW_KEY_3 && action == GLFW_PRESS) {
2436   animacion_camion ^= true;
2437   giroLlanta = 0.0f;
2438   movCamion_x = 118.0f;
2439   movCamion_y = 0.0f;
2440   movCamion_z = 115.0f;
2441
2442   movCamionLuz_x = -3.0f;
2443   movCamionLuz2_x = 3.0f;
2444   movCamionLuz_z = -6.0f;
2445   movCamionLuz2_z = -6.0f;
2446
2447   movCamionLuz_z = movCamion_z - 6.0f;
2448   orientaCamion = 180.0f;
2449   estado_camion = 1;
2450 }
```

Una vez presionada la tecla, procedemos a iniciar la animación, que consta de 12 estados. En esos estados, salvo en los estados que salta, vemos que hay iluminación de tipo puntual en los faros del camión.

**Once the key is pressed, we proceed to start the animation, which consists of 12 states. In those states, except the ones which involve jumps, we see that there is a point light in the truck's headlights.**

```

437 // Para gato camion (Animación 3)
438 if (animacion_camion) {
439     giroLlanta += 0.2f;
440     switch (estado_camion) {
441         case 1:
442             if (movCamion_z >= 70.0f) {
443                 giroLlanta += 1.0f;
444                 movCamion_z -= 1.0f;
445                 movCamionLuz_z = -6.0f;
446                 movCamionLuz2_z = -6.0f;
447                 ilumCamionR = 1.0f;
448                 ilumCamionG = 1.0f;
449                 ilumCamionB = 1.0f;
450             }
451             else {
452                 estado_camion++;
453             }
454             break;
455         case 2: //Salto 1
456             if (movCamion_z >= 60.0f) {
457                 giroLlanta += 0.1f;
458                 movCamion_z -= 1.3;
459                 movCamion_y += 0.3f;
460                 ilumCamionR = 0.0f;
461                 ilumCamionG = 0.0f;
462                 ilumCamionB = 0.0f;
463             }
464             else {
465                 estado_camion++;
466             }
467             break;
468         case 3:
469             if (movCamion_z >= -60.0f) {
470                 giroLlanta += 1.0f;
471                 movCamion_z -= 1.0f;
472                 ilumCamionR = 1.0f;
473                 ilumCamionG = 1.0f;
474                 ilumCamionB = 1.0f;
475                 if (movCamion_y > 0.0f)
476                     movCamion_y -= 0.3f;
477                 else
478                     movCamion_y = 0.0f;
479             }
480             else {
481                 estado_camion++;
482                 movCamion_y = 0.0f;
483             }
484             break;
485         case 4: //Salto 2
486             if (movCamion_z >= -70.0f) {
487                 giroLlanta += 0.1f;
488                 movCamion_z -= 1.3;
489                 movCamion_y += 0.3f;
490                 ilumCamionR = 0.0f;
491                 ilumCamionG = 0.0f;
492                 ilumCamionB = 0.0f;
493             }
494             else {
495                 estado_camion++;
496             }
497             break;

```

```

498         case 5:
499             if (movCamion_z >= -107.0f) {
500                 giroLlanta += 1.0f;
501                 movCamion_z -= 1.0f;
502                 movCamionLuz_z = -6.0f;
503                 ilumCamionR = 1.0f;
504                 ilumCamionG = 1.0f;
505                 ilumCamionB = 1.0f;
506                 if (movCamion_y > 0.0f)
507                     movCamion_y -= 0.3f;
508                 else
509                     movCamion_y = 0.0f;
510             }
511             else {
512                 estado_camion++;
513                 movCamion_y = 0.0f;
514                 orientaCamion = -90.0f;
515                 movCamion_z = -107.0f;
516             }
517             break;
518         case 6: //Arriba de estación
519             if (movCamion_x >= -7.0f) {
520                 giroLlanta += 1.0f;
521                 movCamion_x -= 1.0f;
522                 movCamionLuz_x = -6.0f;
523                 movCamionLuz2_x = -6.0f;
524                 movCamionLuz_z = -3.0f;
525                 movCamionLuz2_z = 3.0f;
526                 ilumCamionR = 1.0f;
527                 ilumCamionG = 1.0f;
528                 ilumCamionB = 1.0f;
529             }
530             else {
531                 estado_camion++;
532                 orientaCamion = 0.0f;
533                 movCamion_x = -7.0f;
534                 movCamionLuz_x = -3.0f;
535                 movCamionLuz2_x = 3.0f;
536             }
537             break;
538         case 7:
539             if (movCamion_z <= -70.0f) {
540                 giroLlanta += 1.0f;
541                 movCamion_z += 1.0f;
542                 movCamionLuz_z = 6.0f;
543                 movCamionLuz2_z = 6.0f;
544                 ilumCamionR = 1.0f;
545                 ilumCamionG = 1.0f;
546                 ilumCamionB = 1.0f;
547             }
548             else {
549                 estado_camion++;
550             }
551             break;
552         case 8: //Salto 3
553             if (movCamion_z <= -60.0f) {
554                 giroLlanta += 0.1f;
555                 movCamion_z += 1.3;
556                 movCamion_y += 0.3f;
557                 ilumCamionR = 0.0f;
558                 ilumCamionG = 0.0f;
559                 ilumCamionB = 0.0f;

```

```

560 } else { ilumCamionB = 0.0f;
561     estado_camion++;
562 }
563 }
564 break;
565 case 9:
566 if (movCamion_z <= 60.0f) {
567     girollanta += 1.0f;
568     movCamion_z += 1.0f;
569     movCamionLuz_z = 6.0f;
570     movCamionLuz2_z = 6.0f;
571     ilumCamionR = 1.0f;
572     ilumCamionG = 1.0f;
573     ilumCamionB = 1.0f;
574     if (movCamion_y > 0.0f)
575         movCamion_y -= 0.3f;
576     else
577         movCamion_y = 0.0f;
578 }
579 else {
580     estado_camion++;
581     movCamion_y = 0.0f;
582 }
583 break;
584 case 10: //Salto 4
585 if (movCamion_z <= 70.0f) {
586     girollanta += 0.1f;
587     movCamion_z += 1.3;
588     movCamion_y += 0.3f;
589     ilumCamionR = 0.0f;
590     ilumCamionG = 0.0f;
591     ilumCamionB = 0.0f;
592 }
593 else {
594     estado_camion++;
595 }
596 break;
597 case 11:
598 if (movCamion_z <= 115.0f) {
599     girollanta += 1.0f;
600     movCamion_z += 1.0f;
601     movCamionLuz_z = 6.0f;
602     movCamionLuz2_z = 6.0f;
603     ilumCamionR = 1.0f;
604     ilumCamionG = 1.0f;
605     ilumCamionB = 1.0f;
606     if (movCamion_y > 0.0f)
607         movCamion_y -= 0.3f;
608     else
609         movCamion_y = 0.0f;
610 }
611 else {
612     estado_camion++;
613     movCamion_y = 0.0f;
614     orientaCamion = 90.0f;
615 }
616 break;
617 case 12: //Hacia la derecha
618 if (movCamion_x <= 118.0f) {
619     girollanta += 1.0f;
620     movCamion_x += 1.0f;
621     movCamionLuz_z = -3.0f;
622     movCamionLuz2_z = 3.0f;
623     movCamionLuz_x = 6.0f;
624     movCamionLuz2_x = 6.0f;
625     ilumCamionR = 1.0f;
626     ilumCamionG = 1.0f;
627     ilumCamionB = 1.0f;
628 }
629 else {
630     estado_camion = 1;
631     orientaCamion = 180.0f;
632     movCamion_x = 118.0f;
633     movCamionLuz_x = -3.0f;
634     movCamionLuz2_x = 3.0f;
635 }
636 break;
637 }
638 }

```

Para la animación 4, la del ovni, tenemos un total de 7 estados. Esta animación involucra iluminación de tipo spotlight. Se enciende con la tecla 4 y se reinicia con la tecla O. En cualquier caso, el ovni siempre estará rotando.

For animation 4, the UFO animation, we have a total of 7 states. This animation involves a spotlight. It is turned on with the key 4 and restarted with the O key. In any case, the UFO will always be rotating.

```
2452      //Animación 4: Secuestro de Futaba
2453      if (key == GLFW_KEY_4 && action == GLFW_PRESS) {
2454          animacion_ovni ^= true;
2455      }
```

```
2456      //Uso una tecla diferente para reiniciarlo
2457      if (key == GLFW_KEY_O && action == GLFW_PRESS) {
2458          animacion_ovni = false;
2459          orientaOvni = 0.0f;
2460          movOvni_x = 10.0f;
2461          movOvni_y = 30.0f;
2462          movOvni_z = 60.0f;
2463          estado_Ovni = 0;
2464          escalaFutaba1 = 0.0f;
2465          escalaFutaba2 = 0.35f;
2466          movFutaba_y = -0.5;
2467          contOvni = 0;
2468          ilumOvni = 0.0f;
2469      }
```

```
640      // Para ovni (Animación 4)
641      if (animacion_ovni) {
642          orientaOvni -= 1.0f;
643          switch (estado_Ovni) {
644              case 0:
645                  if (movOvni_z >= 85) {
646                      estado_Ovni = 1;
647                  }
648                  else {
649                      movOvni_x += 0.02;
650                      movOvni_z += 0.1;
651                  }
652                  break;
653              case 1:
654                  //Desaparece Futaba 2 y aparece Futaba 1
655                  escalaFutaba2 = 0.0f;
656                  escalaFutaba1 = 0.35f;
657                  estado_Ovni = 2;
658                  ilumOvni = 1.0f; //Encendemos luz de ovni
659                  break;
660              case 2:
661                  //Futaba 1 viaja hacia arriba con escala, rotación y traslación en Y
662                  if (escalaFutaba1 >= 0) {
663                      //Movimiento y decremento
664                      escalaFutaba1 -= 0.001;
665                      movFutaba_y += 0.1;
666                  }
667                  else {
668                      estado_Ovni = 3;
```

```

669         }
670         break;
671     case 3:
672         //Pequeño delay xD
673         if (contOvni <= 120) {
674             contOvni++;
675             ilumOvni = 0.0f; //Apagamos luz de ovni
676         }
677         else {
678             ilumOvni = 1.0f; //Encendemos luz de ovni
679             estado_Ovni = 4;
680             contOvni = 0;
681         }
682         break;
683     case 4:
684         //Futaba hacia abajo con escala, rotación y traslación en Y
685         if (escalaFutaba1 < 0.35) {
686             //Movimiento y decremeneto
687             escalaFutaba1 += 0.001;
688             movFutaba_y -= 0.1;
689         }
690         else {
691             estado_Ovni = 5;
692         }
693         break;
694     case 5:
695         //Futaba 1 desaparece y aparece Futaba 1
696         escalaFutaba2 = 0.35f;
697         escalaFutaba1 = 0.0f;
698         estado_Ovni = 6;

```

```

699         ilumOvni = 0.0f; //Apagamos luz
700         break;
701     case 6:
702         //Ovni regresa a posición de inicio
703         if (movOvni_z <= 60) {
704             estado_Ovni = 7;
705         }
706         else {
707             movOvni_x -= 0.02;
708             movOvni_z -= 0.1;
709         }
710         break;
711     }
712 }
713 }
714 else {
715     orientaOvni -= 0.2f; //Garantiza que el ovni siempre esté en movimiento
716 }

```

En el caso de la animación 5, tenemos más complejidad, pues se realiza por keyframes. Para esto, se necesitan varias funciones. Lo primero que mostramos es el funcionamiento: Cuando presionamos la tecla 5, se inicia la animación, empezando por el primer cuadro, interpolamos y avanzamos de cuadro.

**In the case of animation 5, we have more complexity, since it is made with the keyframes method. For this, several functions are needed. The first thing we show is how it works: When**

we press the key 5, the animation starts, starting with the first frame, we interpolate and advance from frame to frame.

```
2471      //Animación 5: Cachetadas a Joker (Keyframes)
2472      if (key == GLFW_KEY_5 && action == GLFW_PRESS)
2473      {
2474          if (play == false && (FrameIndex > 1))
2475          {
2476              std::cout << "Play animation" << std::endl;
2477              resetElements();
2478              //First Interpolation
2479              interpolation();
2480
2481              play = true;
2482              playIndex = 0;
2483              i_curr_steps = 0;
2484          }
2485      }
2486      else
2487      {
2488          play = false;
2489      }
2490 }
```

Lo que hacemos es definir la estructura de nuestros frames, creamos un arreglo de frames que guardará los valores que creemos convenientes y después creamos una función para guardar frames.

**What we do here is to define the structure of our frames, we create an array of frames that will save the values that we think are convenient and then we create a function to save frames.**

```

178     // Definición de frames
179     #define MAX_FRAMES 9
180     int i_max_steps = 60;
181     int i_curr_steps = 0;
182     typedef struct _frame
183     {
184         //Para cachetadas
185         //Joker
186         float giroCabezaJoker_y;
187         //Akechi
188         float giroBrazoAkechi_x;
189         //Ann
190         float giroBrazoDerechoAnn_x;
191         float giroBrazoDerechoAnn_y;
192     }FRAME;
193
194
195     //Arreglo de frames
196     FRAME KeyFrame[MAX_FRAMES];
197     int FrameIndex = 0;
198     bool play = false;
199     int playIndex = 0;
200
201     bool skyboxtype = true;
202
203     void saveFrame(void)
204     {
205         KeyFrame[FrameIndex].giroCabezaJoker_y = giroCabezaJoker_y;
206         KeyFrame[FrameIndex].giroBrazoAkechi_x = giroBrazoAkechi_x;
207         KeyFrame[FrameIndex].giroBrazoDerechoAnn_x = giroBrazoDerechoAnn_x;
208         KeyFrame[FrameIndex].giroBrazoDerechoAnn_y = giroBrazoDerechoAnn_y;
209         FrameIndex++;
210     }

```

Así mismo, tenemos nuestra función para asignar valores ya obtenidos a cada frame:

**Likewise, we have our function to assign values already obtained to each frame:**

```

213     //Valores para nuestra animación por KeyFrames
214     void insertarFrames(void) {
215         //Frame 0:
216         giroCabezaJoker_y = 0.0f;
217         giroBrazoAkechi_x = 0.0f;
218         giroBrazoDerechoAnn_x = 0.0f;
219         giroBrazoDerechoAnn_y = 0.0f;
220         if (FrameIndex < MAX_FRAMES)
221         {
222             saveFrame();
223         }
224         //Frame 1
225         giroCabezaJoker_y = 0.0f;
226         giroBrazoAkechi_x = 0.0f;
227         giroBrazoDerechoAnn_x = -18.6f;
228         giroBrazoDerechoAnn_y = -107.7f;
229         if (FrameIndex < MAX_FRAMES)
230         {
231             saveFrame();
232         }
233         //Frame 2
234         giroCabezaJoker_y = -24.6f;
235         giroBrazoAkechi_x = 0.0f;
236         giroBrazoDerechoAnn_x = 17.1f;
237         giroBrazoDerechoAnn_y = -115.2f;
238         if (FrameIndex < MAX_FRAMES)
239         {
240             saveFrame();
241         }
242         //Frame 3
243         giroCabezaJoker_y = 0.600001f;
244         giroBrazoAkechi_x = 0.0f;
245         giroBrazoDerechoAnn_x = 59.6999f;
246         giroBrazoDerechoAnn_y = -115.2f;
247         if (FrameIndex < MAX_FRAMES)
248         {
249             saveFrame();
250         }
251         //Frame 4
252         giroCabezaJoker_y = 44.9999f;
253         giroBrazoAkechi_x = 0.0f;
254         giroBrazoDerechoAnn_x = 26.1f;
255         giroBrazoDerechoAnn_y = -115.2f;
256         if (FrameIndex < MAX_FRAMES)
257         {
258             saveFrame();
259         }
260         //Frame 5
261         giroCabezaJoker_y = 4.5f;
262         giroBrazoAkechi_x = 0.0f;
263         giroBrazoDerechoAnn_x = -68.6999f;
264         giroBrazoDerechoAnn_y = -47.0999f;
265         if (FrameIndex < MAX_FRAMES)
266         {
267             saveFrame();
268         }
269         //Frame 6
270         giroCabezaJoker_y = 4.49999f;
271         giroBrazoAkechi_x = 172.801f;
272         giroBrazoDerechoAnn_x = -68.6998f;
273         giroBrazoDerechoAnn_y = -5.99983f;
274         if (FrameIndex < MAX_FRAMES)
275         {
276             saveFrame();
277         }

```

```

278         //Frame 7
279         giroCabezaJoker_y = 0.0f;
280         giroBrazoAkechi_x = 0.0f;
281         giroBrazoDerechoAnn_x = 0.0f;
282         giroBrazoDerechoAnn_y = 0.0f;
283         if (FrameIndex < MAX_FRAMES)
284         {
285             saveFrame();
286         }
287     }

```

Una vez que tenemos estos valores, simplemente hay que interpolarlos y reproducirlos.

**Once we have these values, we simply have to interpolate and play them.**

```

297     void interpolation(void)
298     {
299         incGiroCabezaJoker_y = (KeyFrame[playIndex + 1].giroCabezaJoker_y - KeyFrame[playIndex].giroCabezaJoker_y) / i_max_steps;
300         incGiroBrazoAkechi_x = (KeyFrame[playIndex + 1].giroBrazoAkechi_x - KeyFrame[playIndex].giroBrazoAkechi_x) / i_max_steps;
301         incGiroBrazoDerechoAnn_x = (KeyFrame[playIndex + 1].giroBrazoDerechoAnn_x - KeyFrame[playIndex].giroBrazoDerechoAnn_x) / i_max_steps;
302         incGiroBrazoDerechoAnn_y = (KeyFrame[playIndex + 1].giroBrazoDerechoAnn_y - KeyFrame[playIndex].giroBrazoDerechoAnn_y) / i_max_steps;
303     }

```

```

930 //Para cachetadas con keyframes (animación 5)
931 if (play)
932 {
933     if (i_curr_steps >= i_max_steps)
934     {
935         playIndex++;
936         if (playIndex == 3) {
937             //Reproducir sonido 3d
938             morgana->play3D("resources\\sounds\\efectos\\looking-cool-joker.mp3", irrklang::vec3df(13.0f, 1.0f, -30.0f), false, false, false);
939         }
940         if (playIndex > FrameIndex - 2)
941         {
942             playIndex = 0;
943             play = false;
944         }
945         else
946         {
947             i_curr_steps = 0;
948             interpolation();
949         }
950     }
951     else
952     {
953         giroCabezaJoker_y += incGiroCabezaJoker_y;
954         giroBrazoAkechi_x += incGiroBrazoAkechi_x;
955         giroBrazoDerechoAnn_x += incGiroBrazoDerechoAnn_x;
956         giroBrazoDerechoAnn_y += incGiroBrazoDerechoAnn_y;
957         i_curr_steps++;
958     }
959 }

```

Igualmente, como se observa en la última imagen, vemos que hay un sonido de tipo 3D. Esto se reproduce al llegar al 3er frame.

Para la animación 6, activada con la tecla 6, tenemos globos de diálogo que aparecen encima de los personajes. Los globos de diálogo, en esencia, cambian su escala y se elevan en el eje Y. También, se puede reiniciar con la tecla P.

**Likewise, as seen in the last image, we see that there is a 3D sound. This is played upon reaching the 3rd frame.**

**For animation 6, activated with the key 6, we have speech bubbles that appear above the characters. Speech balloons essentially change their scale and rise on the Y axis. Also, the animation can be restarted with the P key.**

```

2491 //Animación de globos
2492 if (key == GLFW_KEY_6 && action == GLFW_PRESS) {
2493     animacion_globos ^= true;
2494     //Reproducir sonido 3d
2495     morgana->play3D("resources\\sounds\\efectos\\looking-cool-joker.mp3", irrklang::vec3df(13.0f, 1.0f, -30.0f), false, false, false);
2496 }
2497
2498 //Uso una tecla diferente para reiniciarlo
2499 if (key == GLFW_KEY_P && action == GLFW_PRESS) {
2500     animacion_globos = false;
2501     eglobo_Joker = 0.0f;
2502     eglobo_Ann = 0.0f;
2503     eglobo_Akechi = 0.0f;
2504     eglobo_Morgana = 0.0f;
2505     estado_globos = 0;
2506     mov_globoY = 0.0f;
2507     mov_globoXZ = 0.0f;
2508 }

```

```

718 // Para globos de dialogo (Animación 6)
719 if (animacion_globos) {
720     switch (estado_globos) {
721         case 0:
722             if (eglobo_Akechi < 1.5f) {
723                 eglobo_Akechi += 0.05;
724                 mov_globoY += 0.01;
725                 mov_globoXZ += 0.01;
726             }
727             else {
728                 estado_globos = 1;
729             }
730             break;
731         case 1:
732             if (eglobo_Akechi > 0.0f) {
733                 eglobo_Akechi -= 0.05;
734             }
735             else {
736                 estado_globos = 2;
737                 mov_globoY = 0.0f;
738                 mov_globoXZ = 0.0f;
739             }
740             break;
741         case 2:
742             if (eglobo_Joker < 1.5f) {
743                 eglobo_Joker += 0.05;
744                 eglobo_Morgana += 0.05;
745                 mov_globoY += 0.01;
746                 mov_globoXZ += 0.01;
747             }
748             else {
749                 estado_globos = 3;
750             }
751             break;
752         case 3:
753             if (eglobo_Joker > 0.0f) {
754                 eglobo_Joker -= 0.05;
755                 eglobo_Morgana -= 0.05;
756             }
757             else {
758                 estado_globos = 4;
759                 mov_globoY = 0.0f;
760                 mov_globoXZ = 0.0f;
761             }
762             break;
763         case 4:
764             if (eglobo_Ann < 1.5f) {
765                 eglobo_Ann += 0.05;
766                 mov_globoY += 0.01;
767                 mov_globoXZ += 0.01;
768             }
769             else {
770                 estado_globos = 5;
771             }
772             break;
773         case 5:
774             if (eglobo_Ann > 0.0f) {
775                 eglobo_Ann -= 0.05;
776             }
777             else {
778                 estado_globos = 6;
779                 mov_globoY = 0.0f;
780                 mov_globoXZ = 0.0f;
781             }
782             break;
783         case 6:
784             if (eglobo_Morgana < 1.5f) {
785                 eglobo_Akechi += 0.05;
786                     eglobo_Morgana += 0.05;
787                     mov_globoY += 0.01;
788                     mov_globoXZ += 0.01;
789                 }
790                 else {
791                     estado_globos = 7;
792                 }
793                 break;
794             case 7:
795                 if (eglobo_Morgana > 0.0f) {
796                     eglobo_Morgana -= 0.05;
797                     eglobo_Akechi -= 0.05;
798                 }
799                 else {
800                     estado_globos = 8;
801                     mov_globoY = 0.0f;
802                     mov_globoXZ = 0.0f;
803                 }
804                 break;
805             case 8:
806                 if (eglobo_Ann < 1.5f) {
807                     eglobo_Ann += 0.05;
808                     mov_globoY += 0.01;
809                     mov_globoXZ += 0.01;
810                 }
811                 else {
812                     estado_globos = 9;
813                 }
814                 break;
815             case 9:
816                 if (eglobo_Ann > 0.0f) {
817                     eglobo_Ann -= 0.05;
818                 }
819                 else {
820                     estado_globos = 10;
821                     mov_globoY = 0.0f;
822                     mov_globoXZ = 0.0f;
823                 }
824                 break;
825             case 10:
826                 if (eglobo_Akechi < 1.5f) {
827                     eglobo_Akechi += 0.05;
828                     mov_globoY += 0.01;
829                     mov_globoXZ += 0.01;
830                 }
831                 else {
832                     estado_globos = 11;
833                 }
834                 break;
835             case 11:
836                 if (eglobo_Akechi > 0.0f) {
837                     eglobo_Akechi -= 0.05;
838                 }
839                 else {
840                     estado_globos = 12;
841                     mov_globoY = 0.0f;
842                     mov_globoXZ = 0.0f;
843                 }
844                 break;
845             case 12:
846                 if (eglobo_Joker < 1.5f) {
847                     eglobo_Joker += 0.05;
848                     mov_globoY += 0.01;
849                     mov_globoXZ += 0.01;
850                 }
851                 else {
852                     estado_globos = 13;
853                 }
854             }
855         }
856     }
857 }

```

```

854     break;
855     case 13:
856         if (eglobo_Joker > 0.0f) {
857             eglobo_Joker -= 0.05;
858         }
859         else {
860             estado_globos = 14;
861             mov_globoY = 0.0f;
862             mov_globoXZ = 0.0f;
863         }
864         break;
865     case 14:
866         if (eglobo_Ann < 1.5f) {
867             eglobo_Morgana += 0.05;
868             eglobo_Ann += 0.05;
869             mov_globoY += 0.01;
870             mov_globoXZ += 0.01;
871         }
872         else {
873             estado_globos = 15;
874         }
875         break;
876     case 15:
877         if (eglobo_Ann > 0.0f) {
878             eglobo_Morgana -= 0.05;
879             eglobo_Ann -= 0.05;
880         }
881         else {
882             estado_globos = 16;
883             mov_globoY = 0.0f;
884             mov_globoXZ = 0.0f;
885         }
886         break;
887     case 16:
888         if (eglobo_Akechi < 1.5f) {
889             eglobo_Akechi += 0.05;
890             mov_globoY += 0.01;
891             mov_globoXZ += 0.01;
892         }
893         else {
894             estado_globos = 17;
895         }
896         break;
897     case 17:
898         if (eglobo_Akechi > 0.0f) {
899             eglobo_Akechi -= 0.05;
900         }
901         else {
902             estado_globos = 18;
903             mov_globoY = 0.0f;
904             mov_globoXZ = 0.0f;
905         }
906         break;
907     case 18:
908         if (eglobo_Morgana < 1.5f) {
909             eglobo_Morgana += 0.05;
910             mov_globoY += 0.01;
911             mov_globoXZ += 0.01;
912         }
913         else {
914             estado_globos = 19;
915         }
916         break;
917     case 19:
918         if (eglobo_Morgana > 0.0f) {
919             eglobo_Morgana -= 0.05;
920         }
921         else {
922             estado_globos = 0;
923             mov_globoY = 0.0f;
924             mov_globoXZ = 0.0f;
925         }
926     break;
927 }
928 }
```

Por último, para la animación 7, activada con la tecla 7 y reiniciada con la tecla L, tenemos a Shadow Morgana corriendo en círculos por todo el cruce. También sube y baja sus brazos para simular el impacto de correr velozmente y apunta a la dirección en que corre. Se basa en fórmulas matemáticas para el recorrido.

**Finally, for animation 7, activated with the key 7 and restarted with the L key, we have Shadow Morgana running in circles around the crossing. It also raises and lowers its arms to simulate the shock of running fast and always looks in the direction where it is running. This animation relies on mathematical formulas.**

```
2510     //Pausa de animación de morgana corriendo
2511     if (key == GLFW_KEY_7 && action == GLFW_PRESS) {
2512         animacion_morgana_corriendo = !animacion_morgana_corriendo;
2513     }
2514
2515     //Usamos L para reiniciar la animación
2516     if (key == GLFW_KEY_L && action == GLFW_PRESS) {
2517         animacion_morgana_corriendo = false;
2518         posMorgana_x = 0.0f;
2519         posMorgana_y = 0.0f;
2520         posMorgana_z = 0.0f;
2521         escMorgana = 1.0f;
2522         giroTorso_y = 0.0f;
2523         giroBrazoMorgana_x = 30.0f;
2524         giroBrazoMorgana_z = 45.0f;
2525         giroBrazoMorganaPositivo = true;
2526     }
2527 }
```

```
961  //Animación de morgana corriendo (Animación 7)
962  if (animacion_morgana_corriendo) {
963      posMorgana_x = 20.0f * cos(glm::radians(giroTorso_y));
964      posMorgana_z = 20.0f * sin(glm::radians(giroTorso_y));
965      giroTorso_y += 1.0f;
966
967      if (giroBrazoMorganaPositivo) {
968          giroBrazoMorgana_z += 3.0f;
969          if (giroBrazoMorgana_z >= 90.0f)
970              giroBrazoMorganaPositivo = false;
971      }
972      else {
973          giroBrazoMorgana_z -= 3.0f;
974          if (giroBrazoMorgana_z <= 45.0f)
975              giroBrazoMorganaPositivo = true;
976      }
977  }
978 }
979 }
```

## Audio / Audio

Para el caso del audio, se usó la librería irrKlang. Lo primero fue descargar los archivos necesarios y añadirlos dentro de las carpetas del proyecto.

**In the case of audio, the irrKlang library was used. The first thing was to download the necessary files and add them inside the project folders.**

Nombre	Fecha de modificación	Tipo	Tamaño
└─\e\venug	29/03/2022 12:41 p. m.	Carpeta de archivos	
└─\include	25/05/2022 09:24 a. m.	Carpeta de archivos	
└─\lib	23/05/2022 12:08 p. m.	Carpeta de archivos	
└─\Modelos_MagicaVoxel	22/05/2022 06:53 p. m.	Carpeta de archivos	
└─\resources	10/05/2022 08:31 p. m.	Carpeta de archivos	
└─\Shaders	23/05/2022 11:20 a. m.	Carpeta de archivos	
└─\assimp-vc141-mtd.dll	25/04/2020 05:07 p. m.	Extensión de la ap...	13,064 KB
└─\Final.cpp	25/05/2022 04:07 p. m.	Archivo de origen ...	83 KB
└─\glad.c	22/04/2020 11:24 p. m.	Archivo de origen C	111 KB
└─\glew32.dll	09/01/2019 09:55 p. m.	Extensión de la ap...	381 KB
└─\glfw3.dll	09/01/2019 09:56 p. m.	Extensión de la ap...	70 KB
└─\ikpFlac.dll	12/02/2018 08:57 a. m.	Extensión de la ap...	156 KB
└─\ikpMP3.dll	12/02/2018 08:57 a. m.	Extensión de la ap...	160 KB
└─\irrKlang.dll	12/02/2018 08:58 a. m.	Extensión de la ap...	524 KB
└─\irrKlangPlayer.exe	12/02/2018 09:11 a. m.	Aplicación	352 KB

Nombre	Fecha de modificación	Tipo	Tamaño
└─\assimp	10/04/2022 05:31 p. m.	Carpeta de archivos	
└─\glad	10/04/2022 05:31 p. m.	Carpeta de archivos	
└─\GLFW_no	10/04/2022 05:31 p. m.	Carpeta de archivos	
└─\glm	23/05/2022 11:00 a. m.	Carpeta de archivos	
└─\irrKlang	06/05/2022 04:22 p. m.	Carpeta de archivos	
└─\KHR	10/04/2022 05:31 p. m.	Carpeta de archivos	
└─\SDL	10/04/2022 05:31 p. m.	Carpeta de archivos	
└─\camera.h	21/05/2022 09:10 p. m.	Archivo de origen ...	5 KB

Nombre	Fecha de modificación	Tipo	Tamaño
└─\assimp-vc141-mtd.dll	25/04/2020 05:07 p. m.	Extensión de la ap...	13,064 KB
└─\assimp-vc141-mtd.exp	25/04/2020 05:07 p. m.	Exports Library File	215 KB
└─\assimp-vc141-mtd.ilk	25/04/2020 05:07 p. m.	Incremental Linker...	34,682 KB
└─\assimp-vc141-mtd.lib	25/04/2020 05:07 p. m.	Object File Library	359 KB
└─\assimp-vc141-mtd.pdb	25/04/2020 05:07 p. m.	Program Debug D...	78,644 KB
└─\glew32.lib	09/01/2019 09:55 p. m.	Object File Library	696 KB
└─\glew32s.lib	09/01/2019 09:55 p. m.	Object File Library	2,387 KB
└─\glfw3.lib	09/01/2019 09:56 p. m.	Object File Library	240 KB
└─\glfw3.dll.lib	09/01/2019 09:56 p. m.	Object File Library	24 KB
└─\irrKlang.exp	12/02/2018 08:58 a. m.	Exports Library File	3 KB
└─\irrKlang.lib	12/02/2018 08:58 a. m.	Object File Library	5 KB

Después, agregamos la librería a nuestro código:

Next, we add the library to our code:

```
32      //Librería de audio:  
33      #if defined(WIN32)  
34          #include <conio.h>  
35      #endif  
36      #include <irrklang/irrklang.h>  
37      #pragma comment(lib, "irrklang.lib") // link with irrKlang.dll
```

Iniciamos los motores de audio, siendo uno dedicado para la música de fondo y el otro para efectos de sonido, y si fallan mandamos un error:

We start the audio engines, one being dedicated for background music and the other for sound effects, and if one engine fails, we send an error:

```
159      //Inicio de audio morgana  
160      irrklang::ISoundEngine* morgana = irrklang::createIrrKlangDevice();  
  
1008      //Inicio de música de fondo  
1009      irrklang::ISoundEngine* bg_music = irrklang::createIrrKlangDevice();  
1010  
1011      if (!bg_music)  
1012          return 0; //Error con la música de fondo  
1013      if (!morgana)  
1014          return 0; //Error con morgana
```

Para el caso de la música de fondo, esta se reproduce al iniciar el renderizado. Mientras tanto, la voz de Morgana se reproduce en ciertos eventos (como animaciones o al presionar la tecla 6).

In the case of background music, it is played when the rendering starts. Meanwhile, Morgana's voice effect plays on certain events (such as animations or pressing the 6 key).

```
1076      //Reproducir música de fondo  
1077      bg_music->play2D("resources\\sounds\\bg_music\\The_Whims_of_Fate.mp3", true);
```

Como el sonido de Morgana es en 3D, se necesita actualizar constantemente la posición desde la que se va a escuchar el sonido:

Since Morgana's sound is in 3D, you need to constantly update the hearing position for the sound to be heard correctly:

```
1203      morgana->setListenerPosition(irrklang::vec3df(camera.Position.x, camera.Position.y, camera.Position.z), irrklang::vec3df(0, 0, 1));
```

Una vez que terminamos el programa, liberamos la memoria de los motores de audio.

Once we end the program, we free the memory of the audio engines.

```
2358      bg_music->drop(); //Borrar música de fondo  
2359      morgana->drop(); //Borrar efecto de sonido de morgana
```