
UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA

INGENIERÍA EN COMPUTACIÓN

LABORATORIO DE COMPUTACIÓN GRÁFICA e INTERACCIÓN HUMANO COMPUTADORA

# REPORTE DE PRÁCTICA Nº 05

**NOMBRE COMPLETO:** Cornejo Aguilar Clara Luz

**Nº de Cuenta:** 316218423

**GRUPO DE LABORATORIO:** 01

**GRUPO DE TEORÍA: 04**

**SEMESTRE 2024-2**

**FECHA DE ENTREGA LÍMITE: 23/03/2024**

**CALIFICACIÓN: _____**

## REPORTE DE PRÁCTICA:

1.- Ejecución de los ejercicios que se dejaron, comentar cada uno y capturas de pantalla de bloques de código generados y de ejecución del programa.

1.- Importar su modelo de coche propio dentro del escenario a una escala adecuada.

Definición de los modelos

```
Model Porsche;
Model Llanta;
Model Cofre;
```

Importando los modelos

```
Porsche = Model();
Porsche.LoadModel("Models/porsche.obj");

Llanta = Model();
Llanta.LoadModel("Models/llanta.obj");

Cofre = Model();
Cofre.LoadModel("Models/cofre.obj");
```

Dibujo de los elementos de forma jerárquica con sus respectivas transformaciones.

```
//Carro
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(-10.0f, 2.0f, -1.5f+mainWindow.getavanza()));
auxCarro = model;
color = glm::vec3(1.0f, 1.0f, 0.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Porsche.RenderModel();

//Cofre
model = auxCarro;
model = glm::translate(model, glm::vec3(0.0f, 0.0f, 7.0f));
model = glm::rotate(model, glm::radians(mainWindow.getcofre()), glm::vec3(1.0f, 0.0f, 0.0f));
color = glm::vec3(1.0f, 0.5f, 0.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Cofre.RenderModel();
```

```
//LLantas
model = auxCarro;
model = glm::translate(model, glm::vec3(3.0f, -2.0f, 6.5f));
model = glm::rotate(model, glm::radians(mainWindow.getllantaAvanza() + mainWindow.getllantaRetrocede()), glm::vec3(1.0f, 0.0f, 0.0f));
color = glm::vec3(0.0f, 0.0f, 0.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Llanta.RenderModel();

model = auxCarro;
model = glm::translate(model, glm::vec3(3.0f, -2.0f, -4.0f));
model = glm::rotate(model, glm::radians(mainWindow.getllantaAvanza()+ mainWindow.getllantaRetrocede()), glm::vec3(1.0f, 0.0f, 0.0f));
color = glm::vec3(0.0f, 0.0f, 0.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Llanta.RenderModel();

model = auxCarro;
model = glm::translate(model, glm::vec3(-3.5f, -2.0f, 6.5f));
model = glm::rotate(model, glm::radians(mainWindow.getllantaAvanza() + mainWindow.getllantaRetrocede()), glm::vec3(1.0f, 0.0f, 0.0f));
color = glm::vec3(0.0f, 0.0f, 0.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Llanta.RenderModel();
```

```
model = auxCarro;
model = glm::translate(model, glm::vec3(-3.5f, -2.0f, -4.0f));
model = glm::rotate(model, glm::radians(mainWindow.getllantaAvanza() + mainWindow.getllantaRetrocede()), glm::vec3(1.0f, 0.0f, 0.0f));
color = glm::vec3(0.0f, 0.0f, 0.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Llanta.RenderModel();
```

2.- Importar sus 4 llantas y acomodarlas jerárquicamente, agregar el mismo valor de rotación a las llantas para que al presionar puedan rotar hacia adelante y hacia atrás.

```
//LLantas
model = auxCarro;
model = glm::translate(model, glm::vec3(3.0f, -2.0f, 6.5f));
model = glm::rotate(model, glm::radians(mainWindow.getllantaAvanza() + mainWindow.getllantaRetrocede()), glm::vec3(1.0f, 0.0f, 0.0f));
color = glm::vec3(0.0f, 0.0f, 0.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Llanta.RenderModel();
```

Se rota cada llanta rotando con las teclas O hacia delante y P hacia atrás.

En Window.h :

```
GLfloat getllantaAvanza() { return llantaAvanza; }
GLfloat getllantaRetrocede() { return llantaRetrocede; }
```

```
GLfloat rotax,rotay,rotaz, pata1, pata2, pata3, pata4, avanza,cofre, llantaAvanza, llantaRetrocede;
```

En Window.cpp

```
llantaAvanza = 0.0f;
llantaRetrocede = 0.0f;
```

```
if (key == GLFW_KEY_O)
{
    theWindow->llantaAvanza += 1.0;
}
if (key == GLFW_KEY_P)
{
    theWindow->llantaRetrocede -= 1.0;
}
```

3.- Importar el cofre del coche, acomodarlo jerárquicamente y agregar la rotación para poder abrir y cerrar.

Para obtener la rotación del cofre utilizamos getcofre()

```
//Cofre
model = auxCarro;
model = glm::translate(model, glm::vec3(0.0f, 0.0f, 7.0f));
model = glm::rotate(model, glm::radians(mainWindow.getcofre()), glm::vec3(1.0f, 0.0f, 0.0f));
color = glm::vec3(1.0f, 0.5f, 0.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Cofre.RenderModel();
```

En Window.h:

```
GLfloat getcofre() { return cofre; }
```

En Window.cpp:

Con las teclas U se abre y con la Y se cierra regresando a su estado original.

```
cofre = 0.0f;

if (key == GLFW_KEY_Y) {
    if (theWindow->cofre< 0) {
        theWindow->cofre += 1.0;
    }
}
if (key == GLFW_KEY_U)
{
    if (theWindow->cofre > -60) {
        theWindow->cofre -= 1.0;
    }
}
```

4.- Agregar traslación con teclado para que pueda avanzar y retroceder de forma independiente

Para obtener su traslación se utiliza getavanza()

```
//Carro
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(-10.0f, 2.0f, -1.5f+mainWindow.getavanza()));
auxCarro = model;
color = glm::vec3(1.0f, 1.0f, 0.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Porsche.RenderModel();
```

En Window.h;

```
GLfloat getavanza() { return avanza; }
```

En Window.cpp:

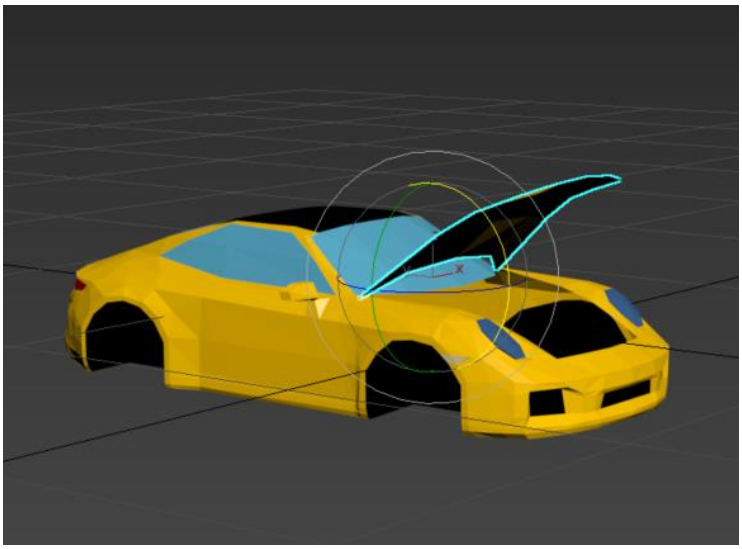
Con la tecla R avanza y con T retrocede.

```
if (key == GLFW_KEY_R) {
    theWindow->avanza += 1.0;
}
if (key == GLFW_KEY_T) {
    theWindow->avanza -= 1.0;
}
```

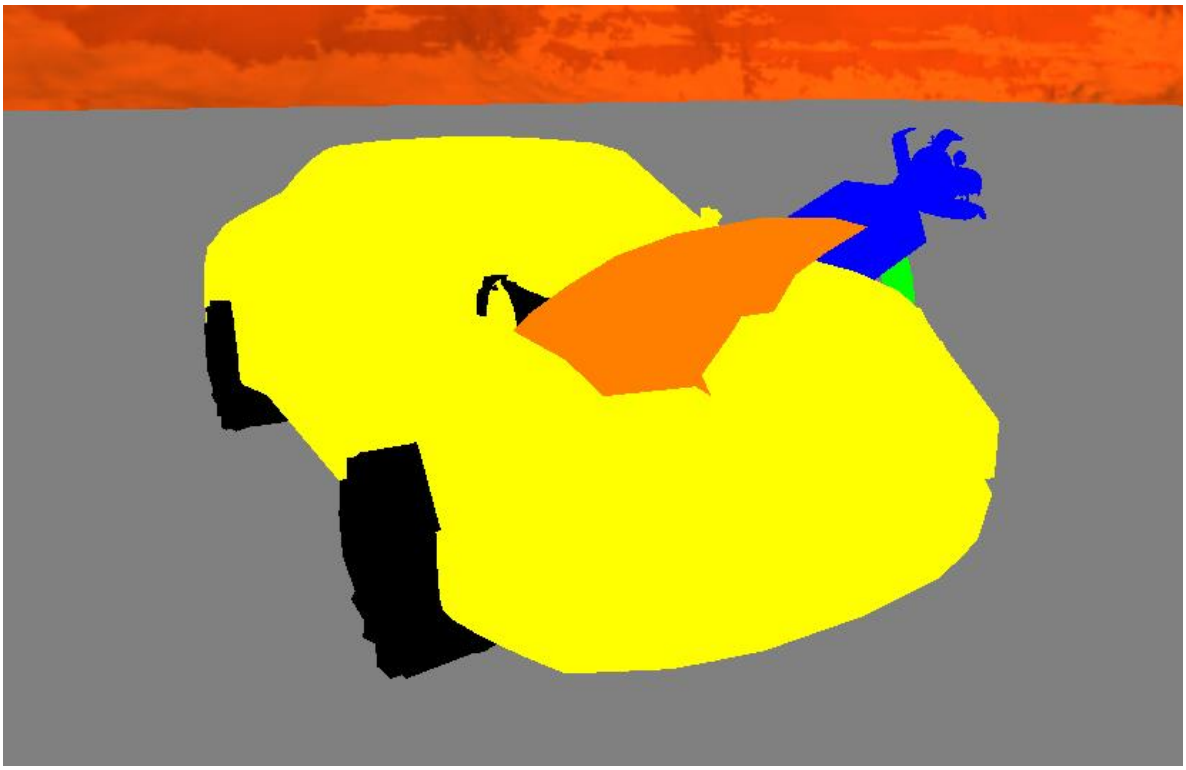
2.- Liste los problemas que tuvo a la hora de hacer estos ejercicios y si los resolvió explicar cómo fue, en caso de error adjuntar captura de pantalla

Tuve problemas al rotar el cofre ya que aunque en 3dMax coloque el pivote en su lugar y rota bien, en el programa su pivote esta en el centro por lo que la rotación se ve mal.

Modelo en 3dMax:



Modelo en el programa:



3.- Conclusión:

La práctica fue una forma muy buena de rectificar los conocimientos adquiridos del modelado jerárquico y sobre todo la importación y adaptación de modelos.

La carga de modelos es fundamental en la computación gráfica ya que para nuestro proyecto es la base para agregar los elementos necesarios para nuestro universo.

Bibliografía en formato APA

*Free Lowpoly 3D - TurboSquid 1359243*. (n.d.). https://www.turbosquid.com/3d-models/car-911-model-4-gts-2017-1359243