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: Tapia Ledesma Angel Hazel

Nº de Cuenta: 320070358

GRUPO DE LABORATORIO: 02

GRUPO DE TEORÍA: 02

SEMESTRE 2026-1

FECHA DE ENTREGA LÍMITE: 27-09-25

CALIFICACIÓN: _____

Actividades realizadas

1.- Importar su modelo de coche propio dentro del escenario a una escala adecuada.
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.
3.- Importar el cofre del coche, acomodarlo jerárquicamente y agregar la rotación para poder abrir y cerrar.
4.- Agregar traslación con teclado para que pueda avanzar y retroceder de forma independiente

Descripción de las actividades

En la primer actividad, lo que se hizo fue tomar el modelo mandado en el previo, en este incluye las texturas, pero en este caso, solo nos importaba el modelo del objeto, en este caso, con terminación .fbx, sin embargo, este modelo era el coche entero, y nosotros necesitábamos el cofre y las llantas por separado, así que se fue a blender, y con lo aprendido en el ejercicio de clase 5, hicimos la separación y la adecuación del origen, después exportamos por separado el carro, después de esto simplemente lo importamos, mediante las variables de modelo creadas en el programa y con la ruta donde se encuentra el archivo de cada una de las partes del auto y finalmente lo escalamos de forma que tuviera cierta coherencia, para esto lo comparamos con goddard del ejercico de clase, para dar una dimensión mayor a él.

En la actividad 2 lo que se realizó fue primero darle la jerarquía de “padre” al auto, de esta manera, utilizamos el modelo del auto cómo matriz auxiliar, en este caso una vez que teníamos el auto, pasamos este modelo auxiliar para colocar las llantas con respecto al auto, en este caso, donde iba el centro de la llanta, también se agregó su articulación respectiva, para darle la rotación a cada llanta, esto se hizo con cada llanta, lo que se hizo fue trasladar cada llanta a su punto adecuado, y en este caso como solo se importaron las llantas trasera y delantera izquierda, se hizo una rotación de 180* para posicionar las del lado derecho, después de ajustar esto, simplemente lo que se hizo fue asignar 2 teclas a la articulación, en este caso, la articulación era la misma, ya que se menciona que todas las llantas debían rotar con las mismas teclas, para esto fuimos a window.cpp e hicimos que hubiera una rotación tanto para “adelante” como para “atrás”, asignando 2 teclas.

Para la actividad 3 lo que se realizó fue similar al punto 2, ya que se tomó la matriz auxiliar del coche para acomodar el cofre con respecto a este. Lo único que cambió para este caso, fue que en la rotación, se limitó el ángulo, ya que hicimos que solo se pudiera

“abrir” el cofre 45 grados, y para cerrarlo regresaba a 0, pero no podía ir a menos grados, ya que atravesaría el carro, esto se hizo igual con 2 teclas en window.cpp .

Para la 4ta actividad, lo que se realizó fue modificar los archivos window.h y window.cpp para agregar una nueva variable “posicionz”, la cual nos ayuda a sumar un valor a la coordenada que recibe el translate en nuestro main. Una vez hecho esto, hicimos la lógica. Lo que se realizó fue que en las mismas teclas del giro de llantas, lo que se hacía era incrementar o decrementar el contador de “posicionz”, incrementar para avanzar y decrementar para retroceder, de esta forma, cuando diéramos la tecla de girar hacia adelante, la posición en Z aumentaría y el carro “avanzaría” y de manera inversa con la otra tecla.

Finalmente, usamos el getter de esta variable “posicionz”, para obtenerla en nuestro main, de forma más específica, esta variable se sumaba a la coordenada en Z de nuestro vector de traslado, y de esta forma, el auto se movía.

Código generado:

Actividad 1:

```
Model Porsche2014Carro_M;
Model Porsche2014Cofre_M;
Model Porsche2014LlantaDelantera_M;
Model Porsche2014LlantaTrasera_M;
```

```
Porsche2014Carro_M = Model();
Porsche2014Carro_M.LoadModel("Models/porsche2014_carro.fbx");

Porsche2014Cofre_M = Model();
Porsche2014Cofre_M.LoadModel("Models/porsche2014_cofre.fbx");

Porsche2014LlantaDelantera_M = Model();
Porsche2014LlantaDelantera_M.LoadModel("Models/porsche2014_llantaDelantera.fbx");

Porsche2014LlantaTrasera_M = Model();
Porsche2014LlantaTrasera_M.LoadModel("Models/porsche2014_llantaTrasera.fbx");
```

```
// Porsche 2014 carro
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.0f, -1.1f, -6.0f + mainWindow.getposicionz()));
model = glm::scale(model, glm::vec3(5.0f, 5.0f, 5.0f));
modelaux = model;
color = glm::vec3(1.0f, 1.0f, 1.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Porsche2014Carro_M.RenderModel();
```

Actividad 2

```
// Articulacion llanta delantera izquierda
model = modelaux;
model = glm::translate(model, glm::vec3(0.74f, 0.17f, 1.17f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion2()), glm::vec3(-1.0f, 0.0f, 0.0f));
model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
// Llanta delantera izquierda
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));
Porsche2014LlantaDelantera_M.RenderModel();

// Articulacion llanta delantera derecha
model = modelaux;
model = glm::translate(model, glm::vec3(-0.74f, 0.17f, 1.17f));
model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion2()), glm::vec3(1.0f, 0.0f, 0.0f));
model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
// Llanta delantera derecha
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));
Porsche2014LlantaDelantera_M.RenderModel();

// Articulacion llanta trasera izquierda
model = modelaux;
model = glm::translate(model, glm::vec3(0.74f, 0.17f, -1.31f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion2()), glm::vec3(-1.0f, 0.0f, 0.0f));
model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
// Llanta trasera izquierda
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));
Porsche2014LlantaTrasera_M.RenderModel();

// Articulacion llanta trasera derecha
model = modelaux;
model = glm::translate(model, glm::vec3(-0.74f, 0.17f, -1.31f));
model = glm::rotate(model, glm::radians(180.0f), glm::vec3(0.0f, 1.0f, 0.0f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion2()), glm::vec3(1.0f, 0.0f, 0.0f));
model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
// Llanta trasera derecha
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));
Porsche2014LlantaTrasera_M.RenderModel();
glUseProgram(0);
```

```
// Patas
// Articulación pata - cuerpo frontal derecha
model = modelaux;
model = glm::translate(model, glm::vec3(-4.0f, -1.2f, -1.5f));
model = glm::rotate(model, glm::radians(30.0f), glm::vec3(0.0f, 0.0f, 0.75f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion4()), glm::vec3(0.0f, 0.0f, 1.0f));
model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
color = glm::vec3(1.0f, 0.71f, 0.75f);
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
//sp.render();

// Muslo frontal derecho
model = glm::translate(model, glm::vec3(0.0f, -1.5f, 0.0f));
modelaux2 = model;
model = glm::scale(model, glm::vec3(0.5f, 3.0f, 1.0f));
color = glm::vec3(1.0f, 0.71f, 0.75f);
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
meshList[2]->RenderMeshGeometry();

// Articulacion rodilla derecha
model = modelaux2;
model = glm::translate(model, glm::vec3(0.0f, -1.5f, 0.0f));
model = glm::rotate(model, glm::radians(-60.0f), glm::vec3(0.0f, 0.0f, 0.75f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion5()), glm::vec3(0.0f, 0.0f, 1.0f));
model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
color = glm::vec3(1.0f, 0.71f, 0.75f);
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
//sp.render();

// Pata frontal derecha
model = glm::translate(model, glm::vec3(0.0f, -1.5f, 0.0f));
modelaux2 = model;
model = glm::scale(model, glm::vec3(0.5f, 3.0f, 1.0f));
color = glm::vec3(1.0f, 0.71f, 0.75f);
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
meshList[2]->RenderMeshGeometry();

// Pie frontal derecho
model = modelaux2;
model = glm::translate(model, glm::vec3(-0.75f, -1.5f, 0.0f));
model = glm::rotate(model, glm::radians(30.0f), glm::vec3(0.0f, 0.0f, 0.75f));
model = glm::scale(model, glm::vec3(1.5f, 0.5f, 1.0f));
color = glm::vec3(0.2f, 0.2f, 0.2f);
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
meshList[2]->RenderMeshGeometry();
```

```
// Articulación pata - cuerpo frontal izquierda
model = modelaux;
model = glm::translate(model, glm::vec3(-4.0f, -1.2f, 1.5f));
model = glm::rotate(model, glm::radians(30.0f), glm::vec3(0.0f, 0.0f, 0.75f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion6()), glm::vec3(0.0f, 0.0f, 1.0f));
model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
color = glm::vec3(1.0f, 0.71f, 0.75f);
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
//sp.render();

// Muslo frontal izquierdo
model = glm::translate(model, glm::vec3(0.0f, -1.5f, 0.0f));
modelaux2 = model;
model = glm::scale(model, glm::vec3(0.5f, 3.0f, 1.0f));
color = glm::vec3(1.0f, 0.71f, 0.75f);
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
meshList[2]->RenderMeshGeometry();

// Articulacion rodilla frontal izquierda
model = modelaux2;
model = glm::translate(model, glm::vec3(0.0f, -1.5f, 0.0f));
model = glm::rotate(model, glm::radians(-60.0f), glm::vec3(0.0f, 0.0f, 0.75f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion7()), glm::vec3(0.0f, 0.0f, 1.0f));
model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
color = glm::vec3(1.0f, 0.71f, 0.75f);
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
//sp.render();

// Pata frontal izquierda
model = glm::translate(model, glm::vec3(0.0f, -1.5f, 0.0f));
modelaux2 = model;
model = glm::scale(model, glm::vec3(0.5f, 3.0f, 1.0f));
color = glm::vec3(1.0f, 0.71f, 0.75f);
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
meshList[2]->RenderMeshGeometry();

// Pie frontal izquierdo
model = modelaux2;
model = glm::translate(model, glm::vec3(-0.75f, -1.5f, 0.0f));
model = glm::rotate(model, glm::radians(30.0f), glm::vec3(0.0f, 0.0f, 0.75f));
model = glm::scale(model, glm::vec3(1.5f, 0.5f, 1.0f));
color = glm::vec3(0.2f, 0.2f, 0.2f);
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
meshList[2]->RenderMeshGeometry();
```

Actividad 3

```
// Articulacion Porshe 2014 cofre
model = modelaux;
model = glm::translate(model, glm::vec3(0.0f, 0.75f, 0.7f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion1()), glm::vec3(1.0f, 0.0f, 0.0f));
model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
// Porsche 2014 cofre
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));
Porsche2014Cofre_M.RenderModel();
```

Actividad 4

```
// Porsche 2014 carro
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.0f, -1.1f, -6.0f + mainWindow.getposicionz()));
model = glm::scale(model, glm::vec3(5.0f, 5.0f, 5.0f));
modelaux = model;
color = glm::vec3(1.0f, 1.0f, 1.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Porsche2014Carro_M.RenderModel();
```

Cómo se utilizó el modelo como padre de los demás elementos, solo hacía falta poner el aumento de posición en Z en este modelo del carro y no en cada pieza.

Archivos modificados para las teclas

Window.h

```h
class Window
{
public:
    Window();
    Window(GLint windowWidth, GLint windowHeight);
    int Initialise();
    GLfloat getBufferWidth() { return bufferWidth; }
    GLfloat getBufferHeight() { return bufferHeight; }
    bool getShouldClose() {
        return  glfwWindowShouldClose(mainWindow);}
    bool* getsKeys() { return keys; }
    GLfloat getXChange();
    GLfloat getYChange();
    void swapBuffers() { return glfwSwapBuffers(mainWindow); }
    GLfloat getrotay() { return rotay; }
    GLfloat getrotax() { return rotax; }
    GLfloat getrotaz() { return rotaz; }
    GLfloat getarticulacion1() { return articulacion1; }
    GLfloat getarticulacion2() { return articulacion2; }
    GLfloat getarticulacion3() { return articulacion3; }
    GLfloat getarticulacion4() { return articulacion4; }
    GLfloat getarticulacion5() { return articulacion5; }
    GLfloat getarticulacion6() { return articulacion6; }
    GLfloat getarticulacion7() { return articulacion7; }
    GLfloat getarticulacion8() { return articulacion8; }
    GLfloat getarticulacion9() { return articulacion9; }
    GLfloat getarticulacion10() { return articulacion10; }
    GLfloat getposicionz() { return posicionz; }

    ~Window();
private:
    GLFWwindow *mainWindow;
    GLint width, height;
    GLfloat rotax,rotay,rotaz, articulacion1, articulacion2, articulacion3, articulacion4, articulacion5, articulacion6, articulacion7, articulacion8, articulacion9, articulacion10, posicionz;
    bool keys[1024];
    GLint bufferWidth, bufferHeight;
    GLfloat lastX;
    GLfloat lastY;
    GLfloat xChange;
    GLfloat yChange;
    bool mouseFirstMoved;
    void createCallbacks();
    static void ManejaTeclado(GLFWwindow* window, int key, int code, int action, int mode);
    static void ManejaMouse(GLFWwindow* window, double xPos, double yPos);
};
```

Window.cpp

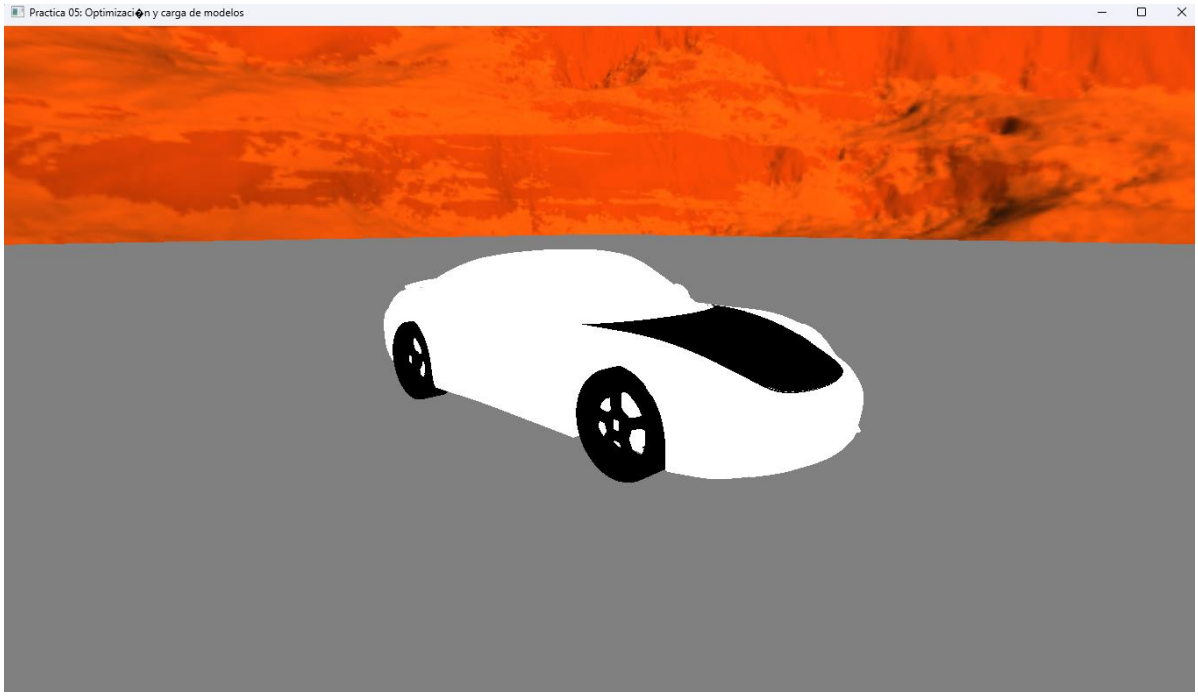
```cpp
Window::Window(GLint windowWidth, GLint windowHeight)
{
    width = windowWidth;
    height = windowHeight;
    rotax = 0.0f;
    rotay = 0.0f;
    rotaz = 0.0f;
    articulacion1 = 0.0f;
    articulacion2 = 0.0f;
    articulacion3 = 0.0f;
    articulacion4 = 0.0f;
    articulacion5 = 0.0f;
    articulacion6 = 0.0f;
    articulacion7 = 0.0f;
    articulacion8 = 0.0f;
    articulacion9 = 0.0f;
    articulacion10 = 0.0f;
    posicionz = 0.0f;

    for (size_t i = 0; i < 1024; i++)
    {
        keys[i] = 0;
    }
}
```

```cpp
if (key == GLFW_KEY_F)
{
    if (theWindow->articulacion1 < 0.0f)
    {
        theWindow->articulacion1 += 10.0f;
    }
}
if (key == GLFW_KEY_G)
{
    if (theWindow->articulacion1 > -45.0f)
    {
        theWindow->articulacion1 -= 10.0f;
    }
}
if (key == GLFW_KEY_H)
{
    theWindow->articulacion2 += 10.0f;
    theWindow->posicionz -= 3.0f;
}
if (key == GLFW_KEY_J)
{
    theWindow->articulacion2 -= 10.0f;
    theWindow->posicionz += 3.0f;
}
```

Ejecución

Actividad 1

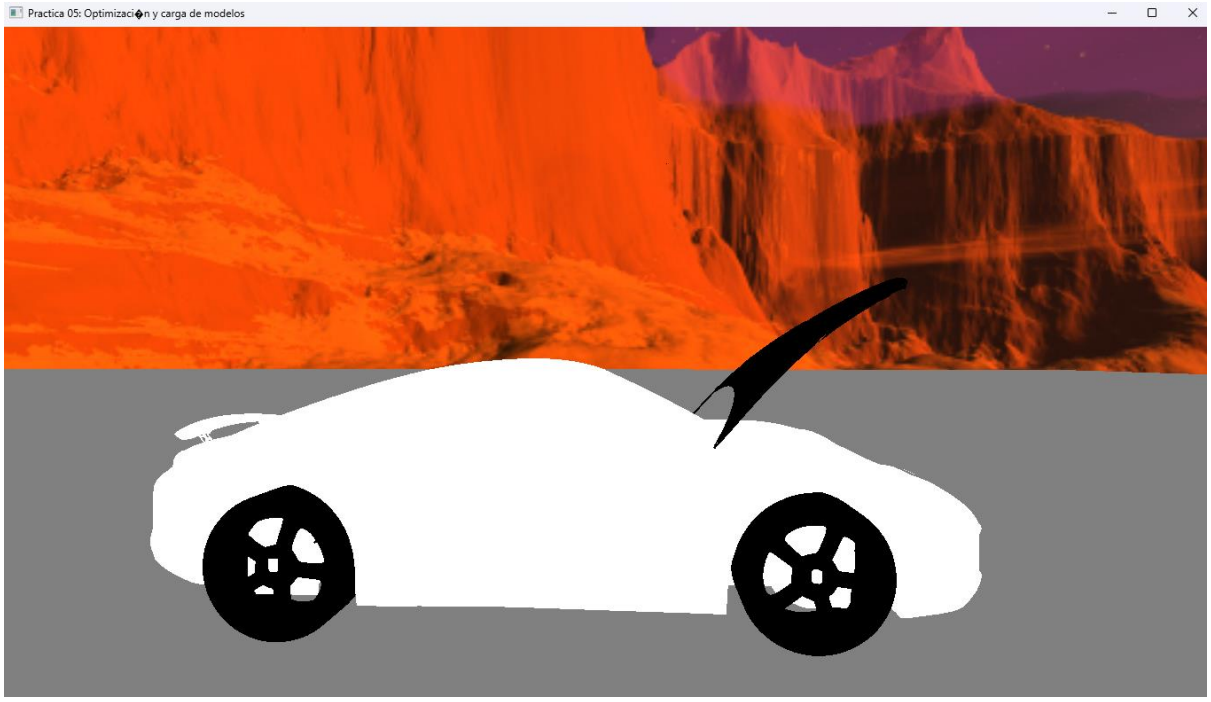


Actividad 2

TECLAS (American Keyboard):

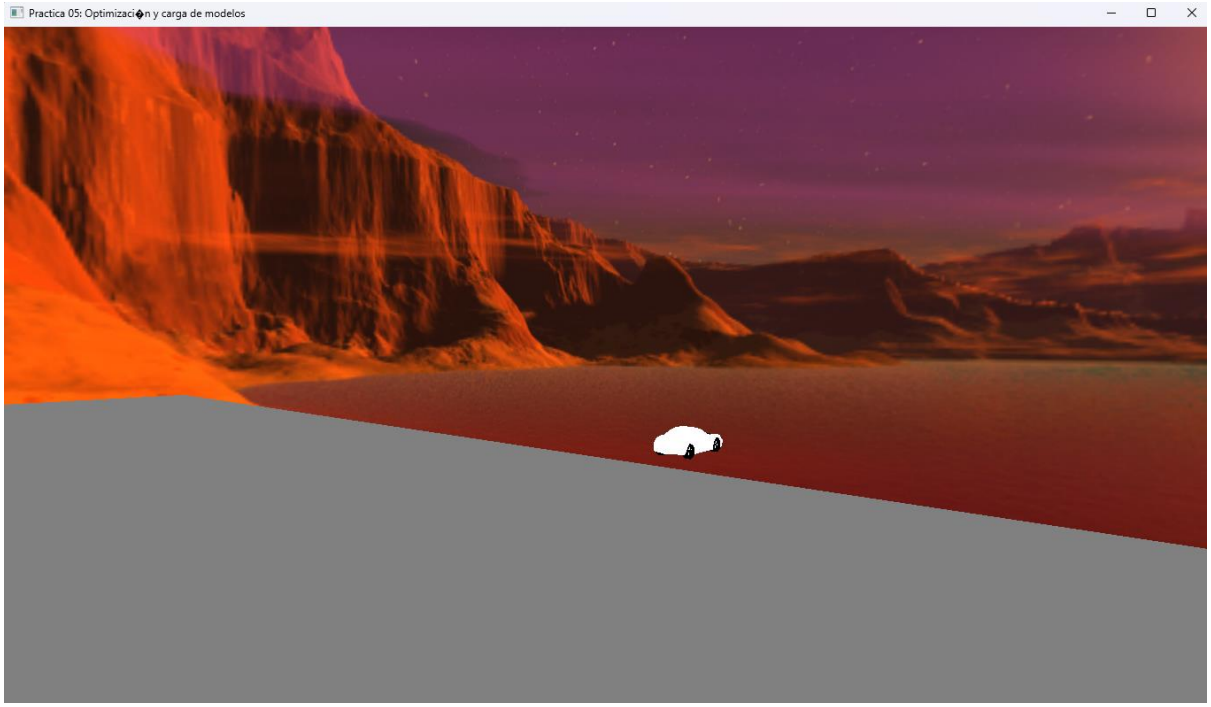
Articulacion llantas (H, J)

Actividad 3
Articulación cofre (F,G)


Practica 05: Optimizaci&#65533;n y carga de modelos

Actividad 4


Practica 05: Optimizaci&#65533;n y carga de modelos



Video

https://youtu.be/LZfBLxl9WnA

Problemas presentados

En esta práctica solo tuve 1 error, el cual fue que cómo expliqué anteriormente, mi posición se realizaba mediante 1 variable, y esta se sumaba en la componente Z del vector de translate, pues bien, puse esta variable sumada a cada vector de traslación de cada elemento del auto, pero al ir adelante o atrás, noté que todas las partes iban más rápido que el auto, pero entre ellas si iban a igual velocidad, al ver bien el código, me di cuenta que, cómo estaba usando la matriz auxiliar como base para la jerarquía de los demás componentes, al aplicar la suma a los demás componentes, realmente se estaba aplicando 2 veces, ya que la traslación del componente base (auto) ya tenía esta traslación, por lo que para resolverlo, borré esta traslación de todos los componentes y solo la dejé en la traslación del objeto que se usó como el padre de la jerarquía, y con esto se resolvió el error.

Conclusión:

La práctica me pareció excelente, creo que ayudó mucho a reforzar lo de importación de modelos y jerarquía, además de que la parte de hacer que el auto se trasladara enfrente o detrás con teclas, es algo nuevo que fue muy interesante pensar en cómo se podría lograr, y nos ayudó a tener más dominio sobre el uso del archivo window.cpp y sobre cómo poder hacer para mover objetos con teclas.

La complejidad me pareció normal, realmente creo que cómo con las practicas pasadas ya se practicó la jerarquía, esto fue muy sencillo, sin embargo, para la parte de traslación, ya era algo nuevo y que tuve que pensar cómo implementarlo, pero tampoco fue algo muy difícil, además de que fue divertido e interesante.

En cuanto a los comentarios, en esta ocasión no surgió comentario alguno, me parece que lo que se tiene que desarrollar está claro, todo se vió en el laboratorio, y en el caso de la traslación, creo que ya teníamos claro que esto se podía realizar con el archivo de window.h y window.cpp.

Bibliografía

- Ddiaz. (2014). “2014 Porsche Cayman S (981)”. Recuperado el 27 de septiembre de 2025 de: https://sketchfab.com/3d-models/2014-porsche-cayman-s-981-03e383400bb74626bcdedbbe198a5849