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: Vázquez Reyes Sebastián

Nº de Cuenta: 318150923

GRUPO DE LABORATORIO: 11

GRUPO DE TEORÍA: 6

SEMESTRE 2024-2

FECHA DE ENTREGA LÍMITE: miércoles 13 de marzo de 2024

CALIFICACIÓN: _____

REPORTE DE PRÁCTICA:

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

El modelo que utilice esta en las referencias de este documento. Afortunadamente este modelo ya tenia separadas las llantas en elementos, solo tuve que separar el cofre para hacer la rotación solicitada mas adelante.

El modelo usado no es el modelo que adjunte en mi previo. La razón del cambio es que el modelo del previo estaba construido como un solo elemento, todo el carro era un solo elemento, y dividir las llantas y el chasis de ese modelo tan complejo seria muy difícil, así que busque uno más sencillo de manejar. Además, este modelo ya también se encontraba a un escalado manejable para el ambiente de OpenGL, el modelo del previo era enorme y también tendría que haberlo hecho más pequeño. El modelo lo importe de la siguiente manera:

```
Model chasis, cofre, llanta1, llanta2, llanta3, llanta4;
```

```
chasis = Model();
cofre = Model();
llanta1 = Model();
llanta2 = Model();
llanta3 = Model();
llanta4 = Model();
chasis.LoadModel("Models/chasis.obj");
cofre.LoadModel("Models/cofre.obj");
llanta1.LoadModel("Models/llanta1.obj");
llanta2.LoadModel("Models/llanta2.obj");
llanta3.LoadModel("Models/llanta3.obj");
llanta4.LoadModel("Models/llanta4.obj");
```

```
//------------*INICIA DIBUJO DEL CARRO-------------------*
//CHASIS
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.0f, 1.0f, -1.5f));
model = glm::rotate(model, glm::radians(mainWindow.getrotay()), glm::vec3(0.0f, 1.0f, 0.0f));
model = glm::translate(model, glm::vec3(0.0, 0.0f, mainWindow.getarticulacion3()));
model = glm::translate(model, glm::vec3(0.0, 0.0f, -mainWindow.getarticulacion4()));
modelaux = model;
color = glm::vec3(0.0f, 0.0f, 1.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
chasis.RenderModel();
```

En el chasis del carro se almacenan las rotaciones para hacerlo girar, la traslación para el punto de origen del modelo, y las traslaciones para hacerlo avanzar y retroceder

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

Para el cofre, lo importe de la siguiente manera:

```
//COFRE
model = modelaux;
model = glm::translate(model, glm::vec3(0.0f, 1.55f, -3.5f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion1()), glm::vec3(-1.0f, 0.0f, 0.0f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion2()), glm::vec3(-1.0f, 0.0f, 0.0f));
color = glm::vec3(0.0f, 1.0f, 0.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
cofre.RenderModel();
```

Para la parte de rotación, utilice las mismas instrucciones que use para el Goddard. En esta parte del código, el eje sobre el que rota el chasis está en negativo porque cuando estaba en positivo, rotaba hacia abajo en vez de hacia arriba, como se supone debe de hacerlo un cofre de carro. Únicamente cambie los límites inferior y superior que podía alcanzar el cofre, ahora alcanza un máximo de 60º y un mínimo de 0º.

```
if (key == GLFW_KEY_F)
{
    if (angulos > -60.0) {
        theWindow->articulacion1 -= 5.0;
        angulos -= 5.0;
    }
}

if (key == GLFW_KEY_G)
{
    if (angulos < 0.0) {
        theWindow->articulacion2 += 5.0;
        angulos += 5.0;
    }
}
```

1.3.- 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.

Para las llantas, fue mas de lo mismo, únicamente trasladarlos y construirlos. Sin embargo, no pude implementar las rotaciones de las llantas dentro del modelado jerárquico. Cuando guardaba las rotaciones de la primera llanta y las pasaba a las demás, solo la primera llanta rotaba sobre su propio centro, el resto lo hacia tomando como punto de referencia el centro de la primera llanta, y giraban alrededor de ella. Debido a esto, decidí dejar del modelado jerárquico las rotaciones de la llanta, e implementar las rotaciones en cada una de ellas. Tiene una rotación positiva para ir adelante y una negativa para ir hacia atrás.

```
//LLANTA 1
model = modelaux;
model = glm::translate(model, glm::vec3(-2.0f, 0.0f, -3.3f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion5()), glm::vec3(1.0f, 0.0f, 0.0f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion6()), glm::vec3(-1.0f, 0.0f, 0.0f));
color = glm::vec3(1.0f, 0.0f, 0.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
llanta1.RenderModel();

//LLANTA 2
model = modelaux;
model = glm::translate(model, glm::vec3(-2.0f, 0.0f, +3.1f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion5()), glm::vec3(1.0f, 0.0f, 0.0f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion6()), glm::vec3(-1.0f, 0.0f, 0.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
llanta2.RenderModel();

// LLANTA 3
model = modelaux;
model = glm::translate(model, glm::vec3(2.0f, 0.0f, -3.3f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion5()), glm::vec3(1.0f, 0.0f, 0.0f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion6()), glm::vec3(-1.0f, 0.0f, 0.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
llanta3.RenderModel();

// LLANTA 4
model = modelaux;
model = glm::translate(model, glm::vec3(2.0f, 0.0f, +3.1f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion5()), glm::vec3(1.0f, 0.0f, 0.0f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion6()), glm::vec3(-1.0f, 0.0f, 0.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
llanta4.RenderModel();
```

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

Con las teclas H y J el auto avanza y retrocede respectivamente.

```
if (key == GLFW_KEY_H)
{
    theWindow->articulacion3 += 0.2;
}
if (key == GLFW_KEY_J)
{
    theWindow->articulacion4 += 0.2;
}
```

Esto es gracias a las ultimas 2 instrucciones de traslación que están en el chasis:

```
//------------*INICIA DIBUJO DEL CARRO-----------------*
//CHASIS
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.0f, 1.0f, -1.5f));
model = glm::rotate(model, glm::radians(mainWindow.getrotay()), glm::vec3(0.0f, 1.0f,
model = glm::translate(model, glm::vec3(0.0, 0.0f, mainWindow.getarticulacion3()));
model = glm::translate(model, glm::vec3(0.0, 0.0f, -mainWindow.getarticulacion4()));
modelaux = model;
color = glm::vec3(0.0f, 0.0f, 1.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
chasis.RenderModel();
```

Cada que se presione la tecla H o J se avanzará o retrocederá una cantidad. Este movimiento se hace a través de una traslación que toma como valor el que devuelve articulacion3 y articulacion4. Además, si se rota sobre el eje Y presionando la tecla R, se puede escoger la dirección sobre la que avance y retroceda el carro.

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

Esta vez no hubo problemas, tras construir el Goddard, calcular las rotaciones y traslaciones que se debían hacer con el carro no fueron tanto problema. Sin embargo, el único problema que no logre resolver fue hacer que las llantas rotaran hacia adelante o atrás cada vez que el carro se moviera.

3.- Conclusión:

a. Los ejercicios del reporte: Complejidad, Explicación.

Fue un ejercicio interesante. Realmente no hubo muchos cambios a comparación del Goddard salvo la parte de hacer avanzar y retroceder al auto. Me tomo unos minutos analizar la situación y se me ocurrió implementar este requisito de la manera que esta arriba. El resto del ejercicio fue bastante similar, solo que un modelo diferente.

b. Comentarios generales: Faltó explicar a detalle, ir más lento en alguna explicación, otros comentarios y sugerencias para mejorar desarrollo de la práctica

Durante la sesión no hubo problema para entender la importación y exportación de modelos. Sin embargo, si tuve problemas en recordar como separar una parte de un modelo que esta construido como un solo elemento (en este caso, el cofre del carro). Tuve que seleccionar triangulo por triangulo la parte del motor para poder separarla con un detach, afortunadamente eran pocos triángulos los que formaban al chasis. Me gustaría que el manual de esta práctica fuera más minucioso para explicar el detach de un elemento que forma parte de uno más grande.

c. Conclusión

Al finalizar esta práctica aprendí a rotar, trasladar y escalar figuras en un ambiente 3D, también aprendí como interactúan unas con otras al ocupar la misma posición. Además, aprendí a crear fondos para los ambientes 3D mediante trasladado y escalado

Bibliografía

- Astoria. ***Cartoon blue track low poly***. Recuperado el 11 de marzo de 2024 de: https://es.3dexport.com/free-3dmodel-cartoon-blue-track-low-poly-280606.htm