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 09

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 16 de abril de 2024

CALIFICACIÓN: _____

REPORTE DE PRÁCTICA:

1.1.- Su dado de 10 caras cae sobre el piso, gira y cae en un número "random", se repite la tirada al presionar una tecla.

La parte del código que se encarga de mover al dado es la siguiente:

```
//Animacion del dado
if (caer) {
    if (dadocaer < 30) {
        dadocaer += dadocaerOffset * deltaTime;
        dadorodar += dadorodarOffset * deltaTime;
    }
    else {
        caer = false;
        rota = true;
    }
}
else if (rota){
    if (dadomover < 29) {
        dadomover += dadomoverOffset * deltaTime;
        dadorodar += dadorodarOffset * deltaTime;
    }
}

if (mainWindow.getarticulacion3() > 0.0) {
    dadocaer = 0.0f;
    dadorodar = 0.0f;
    dadomover = 0.0f;
    rota = false;
    caer = true;
}
```

Primero, el dado cae hasta tocar el suelo, cuando lo toca, cambia los estados de las variables booleanas para ahora empezar a rodar en el suelo. Se detiene después de una distancia en especifico para detenerse siempre en una misma cara.

El ultimo if, el que contiene a la articulación, es el encargado de reiniciar la animación. Cada vez que presionemos H y la soltemos, la animación se reiniciará.

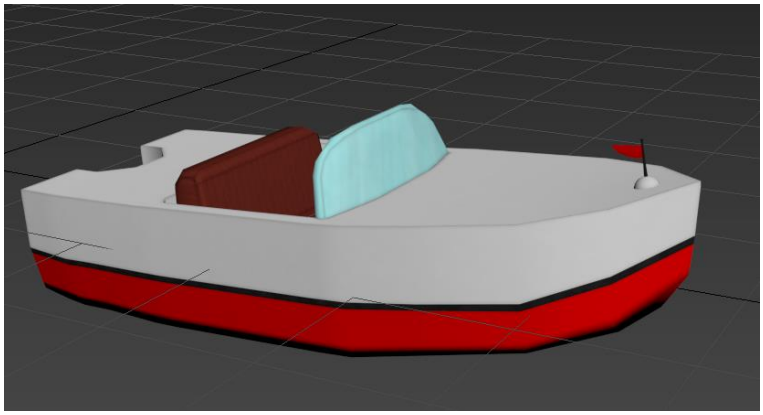
La siguiente parte es el dado importado:

```
//-------------------DIBUJO DEL DADO ------------------------------------
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.0f, 30.0f, 10.0f));
model = glm::translate(model, glm::vec3(0.0f, -dadocaer, 0.0f));
model = glm::translate(model, glm::vec3(0.0f, 0.0f, dadomover));
model = glm::rotate(model, dadorodar * toRadians*6, glm::vec3(1.0f, 0.0f, 0.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
dadoTexture10.UseTexture();
meshList[4]->RenderMesh();
```

1.2.- Por integrante del equipo eligirán un tipo de vehículo: terrestre, aéreo o acuático. Cada integrante del equipo creará un recorrido en el cual el vehículo se moverá y retrocederá haciendo una vuelta en su propio eje al llegar al punto extremo del recorrido. No es necesario que el recorrido abarque toda la extensión del piso

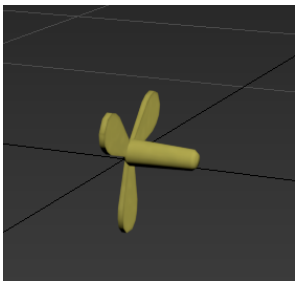
Pertenezco al equipo de Reyes Romero Luis Fernando y González López David. El vehículo que me toco es el acuático.

Para este inciso utilicé el bote móvil común y corriente de Bob Esponja, pero le quite las llantas para que parezca un bote real. El modelo tuve que dividirlo en 3 partes para que presentará una animación correcta y no solo una traslación. El modelo fue dividido en el “chasis” del bote:



El motor que se ubica en la parte trasera:

Y una hélice que impulsa al vehículo:



La parte de código que se encarga de animar a este modelo es la siguiente:

```
//Animacion para mover y rotar el carro
if (avanza) {
    if (movCoche < 100)
    {
        movCoche += movOffset * deltaTime;
        rotllanta += rotllantaOffset * deltaTime;
        traspt2Offset = 0.3f;
    }
    else {
        if (rotpt1 < 180) {
            rotpt1 += rotpt1Offset * deltaTime;
            rotllanta += rotllantaOffset * deltaTime * 0.2;
            traspt2Offset = 0.05f;
        }
        else {
            avanza = false;
        }
    }
} else {
    if (movCoche > -100)
    {
        movCoche -= movOffset * deltaTime;
        rotllanta += rotllantaOffset * deltaTime;
        traspt2Offset = 0.3f;
    }
    else {
        if (rotpt1 > 0) {
            rotpt1 -= rotpt1Offset * deltaTime;
            rotllanta += rotllantaOffset * deltaTime * 0.2;
            traspt2Offset = 0.05f;
        }
        else {
            avanza = true;
        }
    }
}
```

En esta animación, el bote siempre esta haciendo un recorrido de adelante hacia atrás, y cada vez que llega a un extremo del recorrido, da una media vuelta para redirigir su dirección. Las variables movCoche y rotpt1 son las que se encargan de mover al modelo y rotarlo cuando llega a un límite. Dentro de esta misma sección, también se implementa la animación de la hélice, haciéndola girar velozmente cuando avanza el bote, y más lentamente cuando está rotando.

Además, también implemente una animación de “salto” en el motor. Para simular que esta funcionando, el motor está moviéndose de arriba abajo. La siguiente sección de código se encarga de animar esta parte del vehículo.

```
//Animacion del motor del botemovil alzandose y cayendo
if (mover) {
    if (traspt2 < 0.5) {
        traspt2 += traspt2Offset * deltaTime;
    }
    else {
        mover = false;
    }
}
else {
    if (traspt2 > -0.5) {
        traspt2 -= traspt2Offset * deltaTime;
    }
    else {
        mover = true;
    }
}
```

Por último, en la animación completa del vehículo, cada vez que se rota el bote, se modifica el valor de traspt2Offset, para que el motor salte con menor velocidad durante una rotación.

En la siguiente captura se ve el modelo importado con sus animaciones:

```
//-------------------------DIBUJO DEl BOTEMOVIL----------------------------

//Chasis
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(15.0f, 2.0f, -50.0));
model = glm::translate(model, glm::vec3(0.0f, 0.0f, movCoche));
model = glm::rotate(model, rotpt1 * toRadians, glm::vec3(0.0f, 1.0f, 0.0f));
model = glm::scale(model, glm::vec3(0.3f, 0.3f, 0.3f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
botemovilcha.RenderModel();

//Motor trasero
model = glm::translate(model, glm::vec3(-0.6f, -1.0f, -21.0f));
model = glm::translate(model, glm::vec3(0.0f, traspt2, 0.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
botemovilmot.RenderModel();

//Cola del motor
model = glm::translate(model, glm::vec3(-0.2f, -5.7f, -1.6f));
model = glm::rotate(model, rotllanta * toRadians, glm::vec3(0.0f, 0.0f, 1.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
botemovilcol.RenderModel();
```

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

No se me ocurrió una forma de implementar un mecanismo que, mediante números aleatorios generados, pudiera hacer girar al dado en una dirección distinta para que cayera en una cara al azar.

3.- Conclusión:

a. Los ejercicios del reporte: Complejidad, Explicación.

Tener que hacer mucho código spaguetti fue algo relativamente complejo, lo difícil estaba en no perderse en donde abrista el primer if o el else de la lista completa de if´s y else´s. Además, claro, de hacer parar el dado en caras al azar.

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

Gracias a la explicación de la última clase, no hubo problemas para nada.

Conclusión

Al finalizar esta práctica aprendí a implementar animaciones en modelos mediante código. Además, aprendí también a dividir y texturizar modelos descargados.

Bibliografía

- TheoClark. (2022). *Spongebob Boatmobile.* Sketchfab. https://sketchfab.com/3d-models/spongebob-boatmobile-sketchfabweeklychallenge-abdca2ced413486b9c943111621d6f39