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 4

NOMBRE COMPLETO: BRANDON HERNANDEZ SOLIS

Nº de Cuenta: 318263113

GRUPO DE LABORATORIO: 2

GRUPO DE TEORÍA: 6

SEMESTRE 2025-1

FECHA DE ENTREGA LÍMITE: 07/09/2024

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.- Terminar la Grúa con:
- Cuerpo (prisma rectangular)
- Base (pirámide cuadrangular)
- 4 llantas (4 cilindros) con teclado se pueden girar las 4 llantas por separado

Código: Esto solo fue retomando la actividad de clase y añadiendo las líneas de rotate para cada cilindro. Para las articulaciones extra que necesitamos se agregó la declaración en el archivo window.h y se asignó la acción en window.cpp y eso fue suficiente para tener las nuevas articulaciones en las teclas: Z, X, C, V.

Window.h:

```
23      GLfloat getarticulacion1() { return articulacion1; }
24      GLfloat getarticulacion2() { return articulacion2; }
25      GLfloat getarticulacion3() { return articulacion3; }
26      GLfloat getarticulacion4() { return articulacion4; }
27      GLfloat getarticulacion5() { return articulacion5; }
28      GLfloat getarticulacion6() { return articulacion6; }
29      GLfloat getarticulacion7() { return articulacion7; }
30      GLfloat getarticulacion8() { return articulacion8; }
31      GLfloat getarticulacion9() { return articulacion9; }
32      GLfloat getarticulacion10() { return articulacion10; }
33
34      ~Window();
35  private:
36      GLFWwindow *mainWindow;
37      GLint width, height;
38      GLfloat rotax,rotay,rotaz, articulacion1, articulacion2, articulacion3, articulacion4, articulacion5, articulacion6, articulacion7, arti
39      bool keys[1024];
```

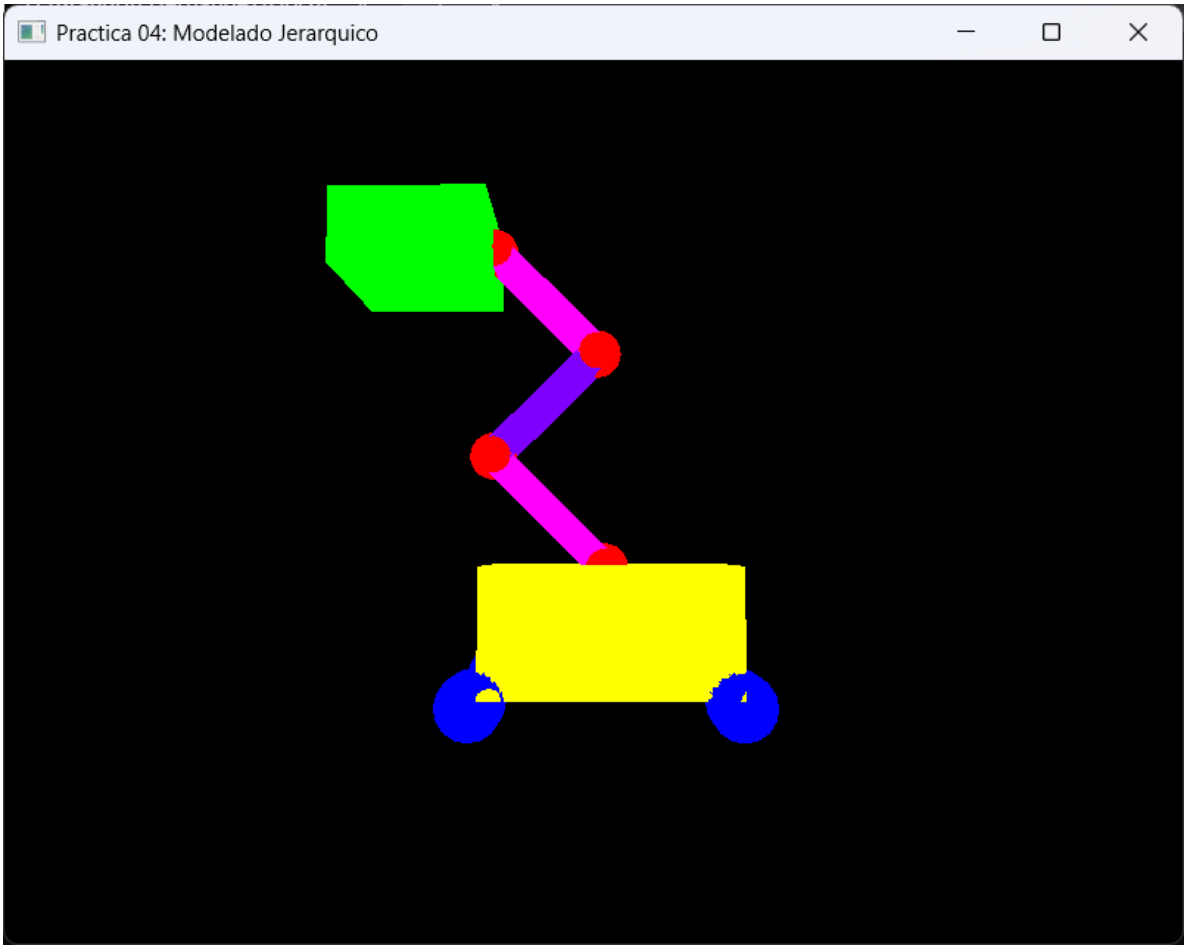
Window.cpp:

```
153     if (key == GLFW_KEY_L)
154     {
155         theWindow->articulacion6 += 10.0;
156     }
157     if (key == GLFW_KEY_Z)
158     {
159         theWindow->articulacion7 += 10.0;
160     }
161     if (key == GLFW_KEY_X)
162     {
163         theWindow->articulacion8 += 10.0;
164     }
165     if (key == GLFW_KEY_C)
166     {
167         theWindow->articulacion9 += 10.0;
168     }
169     if (key == GLFW_KEY_V)
170     {
171         theWindow->articulacion10 += 10.0;
172     }
173
174
175     if (key == GLFW_KEY_D && action == GLFW_PRESS)
```

Main:

```
373         //Posicion Cilindro 1
374         model = glm::translate(model, glm::vec3(4.0f, -2.0f, 3.0f));
375
376         modelaux = model;
377
378         //Cilindro 1
379         model = glm::rotate(model, glm::radians(90.0f), glm::vec3(1.0f, 0.0f, 0.0f));
380         model = glm::rotate(model, glm::radians(mainWindow.getarticulacion7()), glm::vec3(0.0f, 1.0f, 0.0f));   // Rotacion
381         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
382         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
383         color = glm::vec3(0.0f, 0.0f, 1.0f);
384         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
385
386         meshList[2]->RenderMeshGeometry(); //dibuja cubo, pirámide triangular, pirámide base cuadrangular
```

Resultado:



- 2.- Crear un animal robot 3d

- Instanciando cubos, pirámides, cilindros, conos, esferas:

- 4 patas articuladas en 2 partes (con teclado se puede mover las dos articulaciones de cada pata)

- Cola articulada o 2 orejas articuladas. (con teclado se puede mover la cola o cada oreja independiente

Código: Se creo una nueva variable la cual se llama "modelprin" la cual es otra auxiliar que siempre va a estar situada en el origen, esto con el objetivo de facilitar las traslaciones y siempre conservar la escala origen. Para las articulaciones se usaron las declaradas anteriormente en el modelo de la grúa.

```
349         //Posicion Base
350         model = glm::translate(model, glm::vec3(0.0f, 0.0f, -8.0f));
351
352         modelaux = model;   //Modelo Auxiliar
353         modelprin = model;  //Modelo Principal
354
355         // PERRO    ------------------------------------------------------------------------
356
357         //Cuerpo Rectangular
358
359         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
360         model = glm::scale(model, glm::vec3(8.0f, 4.0f, 3.0f));
361         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
362         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
363         color = glm::vec3(0.65f, 0.38f, 0.2f);
364         glUniform3fv(uniformColor, 1, glm::value_ptr(color));   //Color
365         meshList[0]->RenderMesh();  //Dibuja Cubo
366
367         //Cabeza Cubo
368         model = modelprin;
369         model = glm::translate(model, glm::vec3(5.5f, 1.5f, 0.0f));
370         modelaux = model;
371
372         model = glm::scale(model, glm::vec3(3.0f, 3.0f, 3.0f));
373         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
374         color = glm::vec3(0.78f, 0.52f, 0.34f);
375         glUniform3fv(uniformColor, 1, glm::value_ptr(color));   //Color
```

```
388         //Ojos Esferas
389
390         //Izquierdo
391         model = modelaux;
392         model = glm::translate(model, glm::vec3(1.5f, 1.0f, 0.75f));
393
394         model = glm::scale(model, glm::vec3(0.5f, 0.5f, 0.5f));
395         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
396         color = glm::vec3(1.0f, 1.0f, 1.0f);
397         glUniform3fv(uniformColor, 1, glm::value_ptr(color));   //Color
398         sp.render();
399
400         model = glm::translate(model, glm::vec3(0.70f, 0.0f, 0.0f));
401         model = glm::scale(model, glm::vec3(0.5f, 0.5f, 0.5f));
402         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
403         color = glm::vec3(0.32f, 0.24f, 0.06f);
404         glUniform3fv(uniformColor, 1, glm::value_ptr(color));   //Color
405         sp.render();
```

Se decidió añadir orejas rotativas en lugar de la cola articulada:

```
//Orejas Piramides
//Izquierda
model = modelaux;
model = glm::translate(model, glm::vec3(0.0f, 2.25f, 1.0f));
model = glm::scale(model, glm::vec3(1.5f, 1.5f, 1.5f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion1()), glm::vec3(0.0f, -1.0f, 0.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
color = glm::vec3(0.90f, 0.70f, 0.45f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));   //Color
meshList[1]->RenderMesh();  //Dibuja Piramide

//Derecha
model = modelaux;
model = glm::translate(model, glm::vec3(0.0f, 2.25f, -1.0f));
model = glm::scale(model, glm::vec3(1.5f, 1.5f, 1.5f));
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(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
color = glm::vec3(0.90f, 0.70f, 0.45f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));   //Color
meshList[1]->RenderMesh();  //Dibuja Piramide

//Reiniciamos las posiciones
model = modelprin;
modelaux = model;
```

```
//Patas Atras
//Izquierda Arriba
model = modelprin;
model = glm::translate(model, glm::vec3(-2.5f, -0.5f, 2.0f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion3()), glm::vec3(0.0f, 0.0f, -1.0f));
model = glm::translate(model, glm::vec3(0.0f, -0.5f, 0.0f));
modelaux = model;

model = glm::scale(model, glm::vec3(1.0f, 2.0f, 1.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
color = glm::vec3(0.54f, 0.44f, 0.3f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));   //Color
meshList[0]->RenderMesh();  //Dibuja Cubo

//Izquierda Abajo
model = modelaux;
model = glm::translate(model, glm::vec3(0.0f, -1.0f, 0.0f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion4()), glm::vec3(0.0f, 0.0f, -1.0f));
model = glm::translate(model, glm::vec3(0.0f, -1.0f, 0.0f));
modelaux = model;

model = glm::scale(model, glm::vec3(1.0f, 2.0f, 1.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
color = glm::vec3(0.62f, 0.55f, 0.45f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));   //Color
meshList[0]->RenderMesh();  //Dibuja Cubo
```

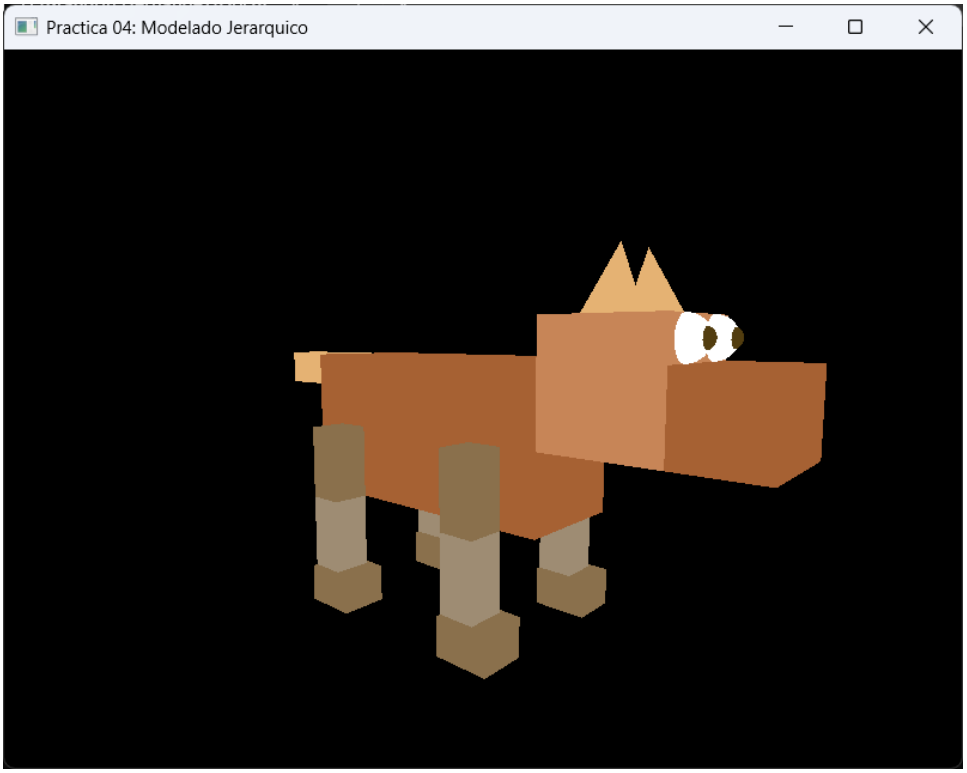
Se agrego una pequeña pieza rectangular al final de cada pata:

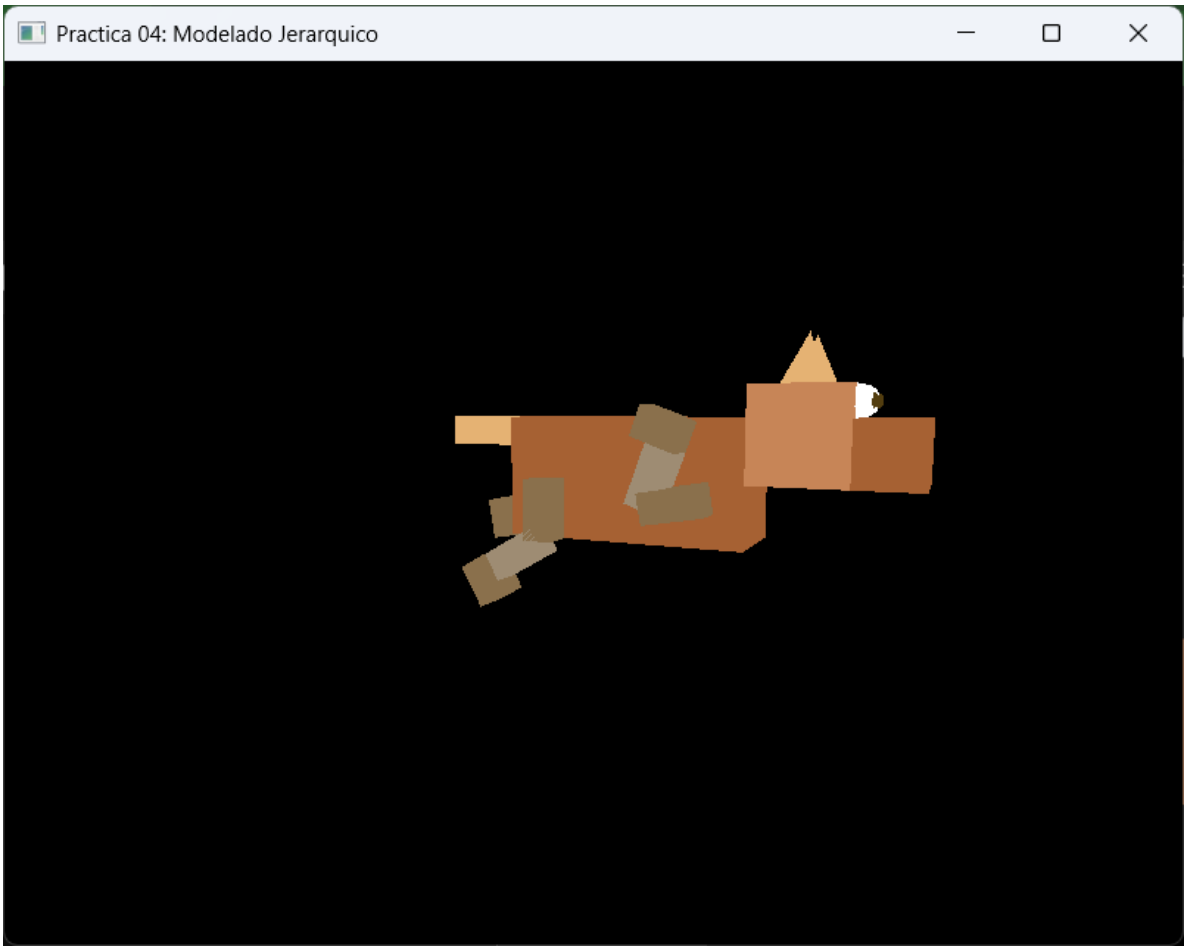
```
//Izquierda Abajo Huella
model = modelaux;
model = glm::translate(model, glm::vec3(0.25f, -1.5f, 0.0f));

model = glm::scale(model, glm::vec3(1.5f, 1.0f, 1.25f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
color = glm::vec3(0.54f, 0.44f, 0.3f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));   //Color
meshList[0]->RenderMesh();  //Dibuja Cubo
```

```
558         //Izquierda Abajo
559         model = modelaux;
560         model = glm::translate(model, glm::vec3(0.0f, -1.0f, 0.0f));
561         model = glm::rotate(model, glm::radians(mainWindow.getarticulacion8()), glm::vec3(0.0f, 0.0f, -1.0f));
562         model = glm::translate(model, glm::vec3(0.0f, -1.0f, 0.0f));
563         modelaux = model;
564
565         model = glm::scale(model, glm::vec3(1.0f, 2.0f, 1.0f));
566         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
567         color = glm::vec3(0.62f, 0.55f, 0.45f);
568         glUniform3fv(uniformColor, 1, glm::value_ptr(color));   //Color
569         meshList[0]->RenderMesh();  //Dibuja Cubo
570
571         //Izquierda Abajo Huella
572         model = modelaux;
573         model = glm::translate(model, glm::vec3(0.25f, -1.5f, 0.0f));
574
575         model = glm::scale(model, glm::vec3(1.5f, 1.0f, 1.25f));
576         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
577         color = glm::vec3(0.54f, 0.44f, 0.3f);
578         glUniform3fv(uniformColor, 1, glm::value_ptr(color));   //Color
579         meshList[0]->RenderMesh();  //Dibuja Cubo
```

```
606         model = glm::scale(model, glm::vec3(1.0f, 2.0f, 1.0f));
607         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
608         color = glm::vec3(0.62f, 0.55f, 0.45f);
609         glUniform3fv(uniformColor, 1, glm::value_ptr(color));   //Color
610         meshList[0]->RenderMesh();  //Dibuja Cubo
611
612         //Izquierda Abajo Huella
613         model = modelaux;
614         model = glm::translate(model, glm::vec3(0.25f, -1.5f, 0.0f));
615
616         model = glm::scale(model, glm::vec3(1.5f, 1.0f, 1.25f));
617         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
618         color = glm::vec3(0.54f, 0.44f, 0.3f);
619         glUniform3fv(uniformColor, 1, glm::value_ptr(color));   //Color
620         meshList[0]->RenderMesh();  //Dibuja Cubo
621
622         //Reiniciamos las posiciones
623         model = modelprin;
624         modelaux = model;
625
```

Resultado:





Teclas usadas:

- F: Oreja derecha
- G: Oreja izquierda
- H: Pata trasera derecha 1
- J: Pata trasera derecha 2
- K: Pata trasera izquierda 1
- L: Pata trasera izquierda 2
- Z: Pata delantera derecha 1
- X: Pata delantera derecha 2
- C: Pata delantera izquierda 1
- V: Pata delantera izquierda 2

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.

Al momento de compilar había veces que no se mostraba correctamente la cámara, como que se iniciaba en otro lugar, pero esto se arreglaba ejecutando otra vez el programa.

3.- Conclusión:

a. Los ejercicios del reporte: Complejidad, Explicación.
b. Comentarios generales: Faltó explicar a detalle, ir más lento en alguna explicación, otros comentarios y sugerencias para mejorar desarrollo de la práctica
c. Conclusión

La práctica estuvo fácil de entender, aprendí mucho sobre el funcionamiento de los modelos en Open GL, además me gusto trabajar con articulaciones y entradas del teclado, creo que fue divertido ver como todo rotaba y arreglar los errores de los modelos. En general, ha sido mi practica favorita porque pude realizarla más fluidamente porque entiendo mejor el código.

Bibliografía en formato APA

- No se uso