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 04

NOMBRE COMPLETO: Miranda González José Francisco

Nº de Cuenta: 318222327

GRUPO DE LABORATORIO: 03

GRUPO DE TEORÍA: 04

SEMESTRE 2025-1

FECHA DE ENTREGA LÍMITE: 07/09/24

CALIFICACIÓN: _____

REPORTE DE PRÁCTICA:

Reporte de Práctica 4: Modelado Jerárquico

Instrucciones:

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

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

Actividades realizadas

Actividad 1

Para completar la primera actividad realice lo siguiente:

Crear dos matrices de modelo 4x4 auxiliares:

```
// Para la grua
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
glm::mat4 modelauxprismarec(1.0);//Inicializar matriz de Modelo 4x4 auxiliar para la jerarquía // para el prisma
glm::mat4 modelauxpiramiderec(1.0);//Inicializar matriz de Modelo 4x4 auxiliar para la jerarquía // para la piramide
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

Una comenzara en el centro del prisma rectangular y la otra en el centro de la pirámide cuadrangular.

Dentro del while de main(), creo el prisma rectangular y guardo su punto central en la matriz 1.

```
//AQUÍ SE DIBUJA LA CABINA, LA BASE, LAS 4 LLANTAS

//prisma rectangular
model = glm::translate(model, glm::vec3(3.0f, 5.0f, -4.0f));//centro prisma rectangular

modelauxprismarec = model;//guarda las transformaciones anteriores

model = glm::scale(model, glm::vec3(6.0f, 2.0f, 2.0f));

glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
color = glm::vec3(1.0f, 0.0f, 1.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
meshList[0]->RenderMesh();

model = modelauxprismarec;// a partir del centro del prisma rectangular (ESTE ES MI INICIO) <----------------------------------
```

A partir de ese punto, me voy al punto central de la pirámide cuadrangular y lo guardo en la matriz 2.

Creo la pirámide cuadrangular.

```
379         //piramide cuadrangular
380         model = glm::translate(model, glm::vec3(0.0f, -2.0f, 0.0f));//centro de la piramide cuadrangular
381
382         modelauxpiramiderec = model; //guarda las transformaciones anteriores
383
384         model = glm::scale(model, glm::vec3(6.0f, 2.0f, 4.0f));// escala piramide
385
386         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
387         color = glm::vec3(0.5f, 0.5f, 0.5f);
388         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
389         meshList[4]->RenderMesh(); //dibuja cubo, pirámide triangular, pirámide base cuadrangular
390         //meshList[2]->RenderMeshGeometry(); //dibuja las figuras geométricas cilindro y cono
391
392         model = modelauxpiramiderec;//a partir del centro de la piramide (ESTE ES MI SEGUNDO INICO) <----------------------------------
```

A partir de ese último punto me traslado y construyo las cuatro llantas.

```
394         //llanta 1. C
395         model = glm::translate(model, glm::vec3(-3.0f, -2.0f, 2.0f));
396         model = glm::rotate(model, glm::radians(90.0f), glm::vec3(1.0f, 0.0f, 0.0f));
397         model = glm::rotate(model, glm::radians(mainWindow.getarticulacionllanta1()), glm::vec3(0.0f, 1.0f, 0.0f));
398
399         model = glm::scale(model, glm::vec3(1.0f, 2.0f, 1.0f));
400
401         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
402         color = glm::vec3(1.0f, 0.18f, 0.2f);
403         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
404         meshList[2]->RenderMeshGeometry();
405
406         model = modelauxpiramiderec;//a partir del centro de la piramide (ESTE ES MI SEGUNDO INICO) <----------------------------------
407
408         //llanta 2. V
409         model = glm::translate(model, glm::vec3(3.0f, -2.0f, 2.0f));
410         model = glm::rotate(model, glm::radians(90.0f), glm::vec3(1.0f, 0.0f, 0.0f));
411         model = glm::rotate(model, glm::radians(mainWindow.getarticulacionllanta2()), glm::vec3(0.0f, 1.0f, 0.0f));
412
413         model = glm::scale(model, glm::vec3(1.0f, 2.0f, 1.0f));
414
415         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
416         color = glm::vec3(1.0f, 0.18f, 0.2f);
417         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
418         meshList[2]->RenderMeshGeometry();
419
420         model = modelauxpiramiderec;//a partir del centro de la piramide (ESTE ES MI SEGUNDO INICO) <----------------------------------
```

```
422         //llanta 3. B
423         model = glm::translate(model, glm::vec3(-3.0f, -2.0f, -2.0f));
424         model = glm::rotate(model, glm::radians(90.0f), glm::vec3(1.0f, 0.0f, 0.0f));
425         model = glm::rotate(model, glm::radians(mainWindow.getarticulacionllanta3()), glm::vec3(0.0f, 1.0f, 0.0f));
426
427         model = glm::scale(model, glm::vec3(1.0f, 2.0f, 1.0f));
428
429         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
430         color = glm::vec3(1.0f, 0.18f, 0.2f);
431         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
432         meshList[2]->RenderMeshGeometry();
433
434         model = modelauxpiramiderec;//a partir del centro de la piramide (ESTE ES MI SEGUNDO INICO) <----------------------------------
435
436         //llanta 4. N
437         model = glm::translate(model, glm::vec3(3.0f, -2.0f, -2.0f));
438         model = glm::rotate(model, glm::radians(90.0f), glm::vec3(1.0f, 0.0f, 0.0f));
439         model = glm::rotate(model, glm::radians(mainWindow.getarticulacionllanta4()), glm::vec3(0.0f, 1.0f, 0.0f));
440
441         model = glm::scale(model, glm::vec3(1.0f, 2.0f, 1.0f));
442
443         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
444         color = glm::vec3(1.0f, 0.18f, 0.2f);
445         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
446         meshList[2]->RenderMeshGeometry();
```

Para hacer que todas las llantas rotaran tuve que modificar los archivos window.cpp y window.h

```
//Para las llantas
////////////////////////////////////////////
articulacionllanta1 = 0.0f;
articulacionllanta2 = 0.0f;
articulacionllanta3 = 0.0f;
articulacionllanta4 = 0.0f;
////////////////////////////////////////////
```

```
//Para las llantas
////////////////////////////////////////////
if (key == GLFW_KEY_C)
{
    theWindow->articulacionllanta1 += 10.0;
}
if (key == GLFW_KEY_V)
{
    theWindow->articulacionllanta2 += 10.0;
}
if (key == GLFW_KEY_B)
{
    theWindow->articulacionllanta3 += 10.0;
}
if (key == GLFW_KEY_N)
{
    theWindow->articulacionllanta4 += 10.0;
}
////////////////////////////////////////////
```

```
//Para las llantas
////////////////////////////////////////////
GLfloat getarticulacionllanta1() { return articulacionllanta1; }
GLfloat getarticulacionllanta2() { return articulacionllanta2; }
GLfloat getarticulacionllanta3() { return articulacionllanta3; }
GLfloat getarticulacionllanta4() { return articulacionllanta4; }
////////////////////////////////////////////
```

```
//Grua y Perro
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
GLfloat rotax,rotay,rotaz, articulacion1, articulacion2, articulacion3, articulacion4, articulacionllanta1, articulacionllanta2, articulacionllanta3,
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

```
////////////////////
 articulacionllanta4,
////////////////////
```

Después de terminar la rotación en cada llanta, regrese al punto central del prisma rectangular para continuar con los brazos de la grúa.

Esta parte ya es idéntica a la del ejercicio de clase.

```
model = modelauxprismarec;// a partir del centro del prisma rectangular (ESTE ES MI INICIO) <----------------------------------

// SE EMPIEZA EL DIBUJO DEL BRAZO

// F G H J
//articulación 1
model = glm::translate(model, glm::vec3(-3.0f, 1.0f, 0.0f)); //aqui empieza la articulación
//rotación alrededor de la articulación que une con la cabina (prisma rectangular)
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion1()), glm::vec3(0.0f, 0.0f, 1.0f));

modelauxprismarec = model;//guarda las transformaciones anteriores

//dibujar una pequeña esfera
//no queremos la informacion de la esfera
model = glm::scale(model, glm::vec3(0.5f, 0.5f, 0.5f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
color = glm::vec3(1.0f, 1.0f, 1.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
sp.render();

model = modelauxprismarec; //a partir de la articulacion 1
```

```
//primer brazo que conecta con la cabina (prisma rectangular)

model = glm::translate(model, glm::vec3(-2.0f, 2.0f, 0.0f));// al centro del brazo 1
model = glm::rotate(model, glm::radians(135.0f), glm::vec3(0.0f, 0.0f, 1.0f)); //a partir de su centro lo rotamos en z

modelauxprismarec = model; //guarda las transformaciones anteriores

model = glm::scale(model, glm::vec3(5.0f, 1.0f, 1.0f));// escala del brazo 1

glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
color = glm::vec3(1.0f, 0.0f, 0.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
meshList[0]->RenderMesh(); //dibuja cubo, pirámide triangular, pirámide base cuadrangular
//meshList[2]->RenderMeshGeometry(); //dibuja las figuras geométricas cilindro y cono

//para descartar la escala que no quiero heredar se carga la información de la matrix auxiliar
model = modelauxprismarec;//a partir del centro del brazo 1
```

```
// los ejes de rotaron al rededor de z
// por eso cambian
//articulación 2
model = glm::translate(model, glm::vec3(2.5f, 0.0f, 0.0f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion2()), glm::vec3(0.0f, 0.0f, 1.0f));//en ese punto rotara la segunda articulacion

modelauxprismarec = model;//guarda las transformaciones anteriores

//dibujar una pequeña esfera
//no queremos la informacion de la esfera
model = glm::scale(model, glm::vec3(0.5f, 0.5f, 0.5f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
color = glm::vec3(1.0f, 1.0f, 1.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
sp.render();

model = modelauxprismarec; //a partir de la articulacion 2

//segundo brazo
model = glm::translate(model, glm::vec3(0.0f, -2.5f, 0.0f));//centro del brazo 2

modelauxprismarec = model;//guarda las transformaciones anteriores

model = glm::scale(model, glm::vec3(1.0f, 5.0f, 1.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
color = glm::vec3(0.0f, 1.0f, 0.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
meshList[0]->RenderMesh(); //dibuja cubo y pirámide triangular

model = modelauxprismarec;// a partir del centro del brazo 2
```

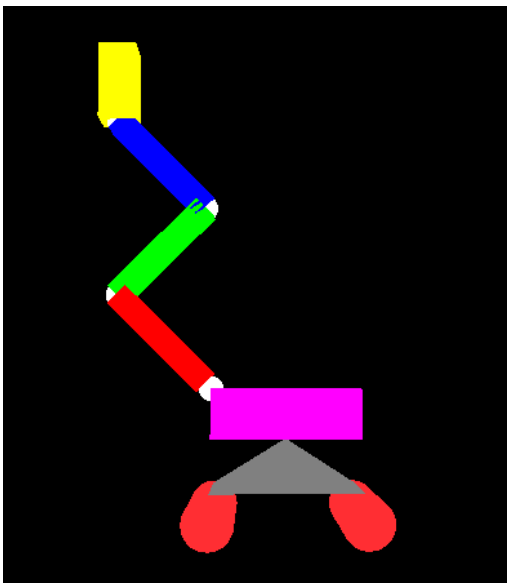
```
//articulación 3 extremo derecho del segundo brazo
model = glm::translate(model, glm::vec3(0.0f, -2.5f, 0.0f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion3()), glm::vec3(0.0f, 0.0f, 1.0f));

modelauxprismarec = model; //guarda las transformaciones anteriores

//dibujar una pequeña esfera
//no queremos la informacion de la esfera
model = glm::scale(model, glm::vec3(0.5f, 0.5f, 0.5f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
color = glm::vec3(1.0f, 1.0f, 1.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
sp.render();

model = modelauxprismarec; //a partir de la articulacion 3
```

```
535         //tercer brazo
536         model = glm::translate(model, glm::vec3(2.5f, 0.0f, 0.0f));//centro del brazo 3
537
538         modelauxprismarec = model;//guarda las transformaciones anteriores
539
540         model = glm::scale(model, glm::vec3(5.0f, 1.0f, 1.0f));
541         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
542         color = glm::vec3(0.0f, 0.0f, 1.0f);
543         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
544         meshList[0]->RenderMesh(); //dibuja cubo y pirámide triangular
545
546         model = modelauxprismarec;// a partir del centro del brazo 3
547
548         //articulación 4
549         model = glm::translate(model, glm::vec3(2.5f, 0.0f, 0.0f));
550         model = glm::rotate(model, glm::radians(45.0f), glm::vec3(0.0f, 0.0f, 1.0f)); //a partir de su centro lo rotamos en z
551         model = glm::rotate(model, glm::radians(mainWindow.getarticulacion4()), glm::vec3(0.0f, 1.0f, 0.0f));
552
553         modelauxprismarec = model; //guarda las transformaciones anteriores
554
555         //dibujar una pequeña esfera
556         //no queremos la informacion de la esfera
557         model = glm::scale(model, glm::vec3(0.5f, 0.5f, 0.5f));
558         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
559         color = glm::vec3(1.0f, 1.0f, 1.0f);
560         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
561         sp.render();
562
563         model = modelauxprismarec; //a partir de la articulacion 4
```

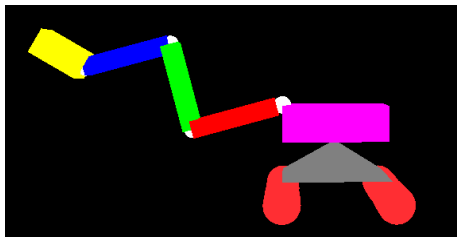
```
565         //canasta
566         model = glm::translate(model, glm::vec3(0.0f, -1.5f, 0.0f));//centro de la canasta
567
568         modelauxprismarec = model;//guarda las transformaciones anteriores
569
570         model = glm::scale(model, glm::vec3(1.5f, 3.0f, 1.5f));
571         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
572         color = glm::vec3(1.0f, 1.0f, 0.0f);
573         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
574         meshList[0]->RenderMesh(); //dibuja cubo y pirámide triangular
575
576         model = modelauxprismarec;// a partir del centro de la canasta
```

Ejecución del programa:

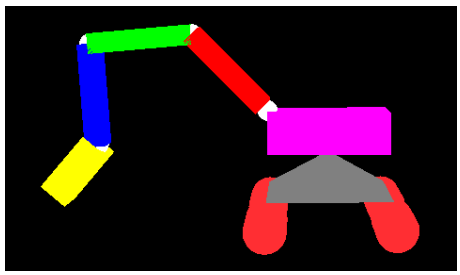
Vista inicial:



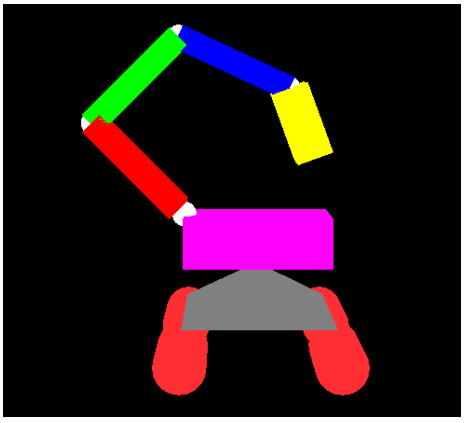
Brazo 1:



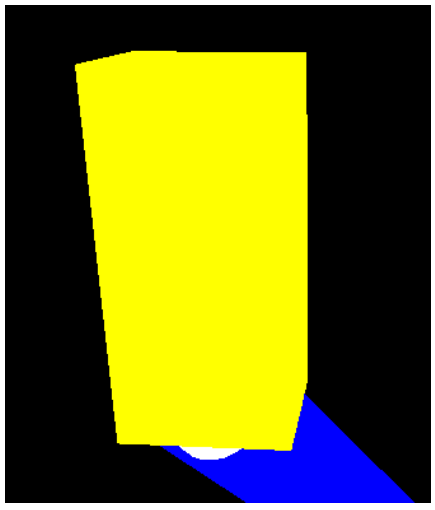
Brazo 2:



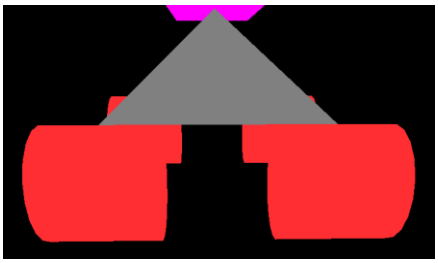
Brazo 3:



Canasta:



Llantas:



En las llantas y la canasta no se logra apreciar la rotación mediante una imagen.

Al ejecutar el programa es más fácil notarlo.

Actividad 2:

Para completar la segunda actividad realice lo siguiente:

Crear siete matrices de modelo 4x4 auxiliares:

```
321     // Para el perro
322     ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
323     glm::mat4 modelaux_Cuerpo_Perro(1.0);
324     glm::mat4 modelaux_Cabeza_Perro(1.0);
325     glm::mat4 modelaux_Cola_Perro(1.0);
326     glm::mat4 modelaux_Pata_Perro_1(1.0);
327     glm::mat4 modelaux_Pata_Perro_2(1.0);
328     glm::mat4 modelaux_Pata_Perro_3(1.0);
329     glm::mat4 modelaux_Pata_Perro_4(1.0);
330     ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

Una comienza en el centro del cuerpo del perro, otra comienza en el centro de la cabeza del perro, otra comienza en el inicio de la cola y las otras cuatro, cada una en el inicio de una pata diferente.

Dentro del while de main(), creo el prisma rectangular (cuerpo del perro) y guardo su punto central en la matriz 1.

```
582         //prisma rectangular (cuerpo del perro)
583         model = glm::translate(model, glm::vec3(0.0f, 7.0f, -6.0f));//centro prisma rectangular (cuerpo del perro)
584
585         modelaux_Cuerpo_Perro = model;//guarda las transformaciones anteriores
586
587         model = glm::scale(model, glm::vec3(12.0f, 6.0f, 6.0f));
588
589         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
590         glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
591         glUniformMatrix4fv(uniformView, 1, GL_FALSE, glm::value_ptr(camera.calculateViewMatrix()));
592         color = glm::vec3(0.65f, 0.32f, 0.16f);
593         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
594         meshList[0]->RenderMesh();
595
596         model = modelaux_Cuerpo_Perro;// a partir del centro del prisma rectangular (cuerpo del perro) (ESTE ES MI INICIO) <----------------------------------
```

A partir de ese punto, me traslado al punto central del prisma rectangular (cabeza) y lo guardo en la matriz 2.

```
598         //prisma rectangular (cabeza)
599         model = glm::translate(model, glm::vec3(-3.0f, 6.0f, 0.0f));//centro del prisma rectangular (cebeza)
600
601         modelaux_Cabeza_Perro = model; //guarda las transformaciones anteriores
602
603         model = glm::scale(model, glm::vec3(6.0f, 6.0f, 6.0f));// escala piramide
604
605         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
606         color = glm::vec3(0.65f, 0.32f, 0.16f);
607         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
608         meshList[0]->RenderMesh(); //dibuja cubo, pirámide triangular, pirámide base cuadrangular
609
610         model = modelaux_Cabeza_Perro;//a partir del centro del prisma rectangular (cabeza) (ESTE ES MI SEGUNDO INICO) <----------------------------------
```

A partir de ese punto, me traslado para construir las orejas, los ojos y el hocico del perro.

```
612         //prisma rectangular (hocico)
613         model = glm::translate(model, glm::vec3(-5.0f, -1.5f, 0.0f));
614         model = glm::rotate(model, glm::radians(90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
615
616         model = glm::scale(model, glm::vec3(1.0f, 3.99f, 2.0f));
617
618         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
619         color = glm::vec3(0.75f, 0.75f, 0.75f);
620         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
621         meshList[2]->RenderMeshGeometry();
622
623         model = modelaux_Cabeza_Perro;//a partir del centro del prisma rectangular (cabeza) (ESTE ES MI SEGUNDO INICO) <----------------------------------
624
625         //prisma rectangular (ojo 1)
626         model = glm::translate(model, glm::vec3(-3.5f, 1.0f, 1.5f));
627
628         model = glm::scale(model, glm::vec3(1.0f, 2.0f, 2.0f));
629
630         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
631         color = glm::vec3(1.0f, 0.0f, 0.0f);
632         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
633         meshList[0]->RenderMesh();
634
635         model = modelaux_Cabeza_Perro;//a partir del centro del prisma rectangular (cabeza) (ESTE ES MI SEGUNDO INICO) <----------------------------------
636
637         //prisma rectangular (ojo 2)
638         model = glm::translate(model, glm::vec3(-3.5f, 1.0f, -1.5f));
639
640         model = glm::scale(model, glm::vec3(1.0f, 2.0f, 2.0f));
641
642         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
643         color = glm::vec3(1.0f, 0.0f, 0.0f);
644         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
645         meshList[0]->RenderMesh();
646
647         model = modelaux_Cabeza_Perro;//a partir del centro del prisma rectangular (cabeza) (ESTE ES MI SEGUNDO INICO) <----------------------------------
```

```
649         //piramide rectangular (oreja 1)
650         model = glm::translate(model, glm::vec3(1.0f, 5.0f, 1.5f));
651
652         model = glm::scale(model, glm::vec3(4.0f, 4.0f, 3.0f));
653
654         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
655         color = glm::vec3(0.75f, 0.75f, 0.75f);
656         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
657         meshList[4]->RenderMesh();
658
659         model = modelaux_Cabeza_Perro;//a partir del centro del prisma rectangular (cabeza) (ESTE ES MI SEGUNDO INICO) <----------------------------------
660
661         //piramide rectangular (oreja 2)
662         model = glm::translate(model, glm::vec3(1.0f, 5.0f, -1.5f));
663
664         model = glm::scale(model, glm::vec3(4.0f, 4.0f, 3.0f));
665
666         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
667         color = glm::vec3(0.75f, 0.75f, 0.75f);
668         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
669         meshList[4]->RenderMesh();
```

Al terminar de construir toda la cabeza, regreso al centro del cuerpo del perro.

A partir de ese punto, me traslado al punto central de la esfera (cola) y lo guardo en la matriz 3.

```
671         model = modelaux_Cuerpo_Perro;// a partir del centro del prisma rectangular (cuerpo del perro) (ESTE ES MI INICIO) <----------------------------------
672
673         //cola esfera
674         model = glm::translate(model, glm::vec3(7.0f, 2.0f, 0.0f)); //aqui empieza la articulación de la cola
675         model = glm::rotate(model, glm::radians(mainWindow.getarticulacioncola()), glm::vec3(1.0f, 0.0f, 0.0f)); // Mover con: P
676
677         modelaux_Cola_Perro = model;//guarda las transformaciones anteriores
678
679         //dibujar una pequeña esfera
680         //no queremos la informacion de la esfera
681         model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
682         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
683         color = glm::vec3(1.0f, 1.0f, 1.0f);
684         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
685         sp.render();
686
687         model = modelaux_Cola_Perro; // a partir de la articulacion de la cola (esfera) (ESTE ES MI TERCER INICIO. 1) <----------------------------------
```

A partir de ese punto, termine de construir la cola con el cilindro y cono.

```
689         //cola cilindro
690         model = glm::translate(model, glm::vec3(0.0f, 3.0f, 0.0f));
691
692         modelaux_Cola_Perro = model;//guarda las transformaciones anteriores
693
694         model = glm::scale(model, glm::vec3(1.0f, 4.0f, 1.0f));
695
696         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
697         color = glm::vec3(0.75f, 0.75f, 0.75f);
698         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
699         meshList[2]->RenderMeshGeometry();
700
701         model = modelaux_Cola_Perro; // a partir de el centro de la cola (cilindro) (ESTE ES MI TERCER INICIO. 2) <----------------------------------
702
703         //cola cono
704         model = glm::translate(model, glm::vec3(0.0f, 3.0f, 0.0f));
705
706         modelaux_Cola_Perro = model;//guarda las transformaciones anteriores
707
708         model = glm::scale(model, glm::vec3(0.5f, 2.0f, 0.5f));
709
710         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
711         color = glm::vec3(1.0f, 0.0f, 0.0f);
712         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
713         meshList[3]->RenderMeshGeometry();
```

Al terminar toda la cola, regreso al centro del cuerpo del perro.

A partir de ese punto, me traslado al punto central de la esfera (pata 1) y lo guardo en la matriz 4.

A partir de ese punto, termine de construir la pata 1 con el cilindro 1, articulación 2 y cilindro 2.

```
715         model = modelaux_Cuerpo_Perro;// a partir del centro del prisma rectangular (cuerpo del perro) (ESTE ES MI INICIO) <----------------------------------
716
717         //pata 1 esfera 1
718         model = glm::translate(model, glm::vec3(-3.0f, 0.0f, 4.0f));
719         model = glm::rotate(model, glm::radians(-45.0f), glm::vec3(0.0f, 0.0f, 1.0f));
720         model = glm::rotate(model, glm::radians(mainWindow.getarticulacionpata1_1()), glm::vec3(0.0f, 0.0f, 1.0f)); // Mover con: U
721
722         modelaux_Pata_Perro_1 = model;//guarda las transformaciones anteriores
723
724         //dibujar una pequeña esfera
725         //no queremos la informacion de la esfera
726         model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
727         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
728         color = glm::vec3(1.0f, 1.0f, 1.0f);
729         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
730         sp.render();
731
732         model = modelaux_Pata_Perro_1;// a partir de la articulacion 1 de la pata 1 (esfera) (ESTE ES MI CUARTO INICIO. 1) <----------------------------------
733
734         //pata 1 cilindro 1
735         model = glm::translate(model, glm::vec3(3.0f, 0.0f, 0.0f));
736         model = glm::rotate(model, glm::radians(-90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
737
738         modelaux_Pata_Perro_1 = model;//guarda las transformaciones anteriores
739
740         model = glm::scale(model, glm::vec3(1.0f, 4.0f, 1.0f));
741
742         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
743         color = glm::vec3(0.75f, 0.75f, 0.75f);
744         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
745         meshList[2]->RenderMeshGeometry();
746
747         model = modelaux_Pata_Perro_1;// a partir del centro de la pata 1 (cilindro) (ESTE ES MI CUARTO INICIO. 2) <----------------------------------
```

```
749         //pata 1 esfera 2
750         model = glm::translate(model, glm::vec3(0.0f, 3.0f, 0.0f));
751         model = glm::rotate(model, glm::radians(-90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
752         model = glm::rotate(model, glm::radians(mainWindow.getarticulacionpata1_2()), glm::vec3(0.0f, 0.0f, 1.0f)); // Mover con: I
753
754         modelaux_Pata_Perro_1 = model;//guarda las transformaciones anteriores
755
756         //dibujar una pequeña esfera
757         //no queremos la informacion de la esfera
758         model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
759         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
760         color = glm::vec3(1.0f, 1.0f, 1.0f);
761         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
762         sp.render();
763
764         model = modelaux_Pata_Perro_1;// a partir de la articulacion 2 de la pata 1 (esfera) (ESTE ES MI CUARTO INICIO. 3) <----------------------------------
765
766         //pata 1 cilindro 2
767         model = glm::translate(model, glm::vec3(0.0f, 2.0f, 0.0f));
768
769         modelaux_Pata_Perro_1 = model;//guarda las transformaciones anteriores
770
771         model = glm::scale(model, glm::vec3(1.0f, 2.0f, 1.0f));
772
773         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
774         color = glm::vec3(0.75f, 0.75f, 0.75f);
775         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
776         meshList[2]->RenderMeshGeometry();
```

Al terminar toda la pata 1, regreso al centro del cuerpo del perro.

Repito el mismo proceso de la pata 1 para las tres patas restantes.

```
778         model = modelaux_Cuerpo_Perro;// a partir del centro del prisma rectangular (cuerpo del perro) (ESTE ES MI INICIO) <----------------------------------
779
780         //pata 2 esfera 1
781         model = glm::translate(model, glm::vec3(3.0f, 0.0f, 4.0f));
782         model = glm::rotate(model, glm::radians(-45.0f), glm::vec3(0.0f, 0.0f, 1.0f));
783         model = glm::rotate(model, glm::radians(mainWindow.getarticulacionpata2_1()), glm::vec3(0.0f, 0.0f, 1.0f)); // Mover con: O
784
785         modelaux_Pata_Perro_2 = model;//guarda las transformaciones anteriores
786
787         //dibujar una pequeña esfera
788         //no queremos la informacion de la esfera
789         model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
790         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
791         color = glm::vec3(1.0f, 1.0f, 1.0f);
792         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
793         sp.render();
794
795         model = modelaux_Pata_Perro_2;// a partir de la articulacion 1 de la pata 2 (esfera) (ESTE ES MI QUINTO INICIO. 1) <----------------------------------
796
797         //pata 2 cilindro 1
798         model = glm::translate(model, glm::vec3(3.0f, 0.0f, 0.0f));
799         model = glm::rotate(model, glm::radians(-90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
800
801         modelaux_Pata_Perro_2 = model;//guarda las transformaciones anteriores
802
803         model = glm::scale(model, glm::vec3(1.0f, 4.0f, 1.0f));
804
805         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
806         color = glm::vec3(0.75f, 0.75f, 0.75f);
807         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
808         meshList[2]->RenderMeshGeometry();
809
810         model = modelaux_Pata_Perro_2;// a partir del centro de la pata 2 (cilindro) (ESTE ES MI QUINTO INICIO. 2) <----------------------------------
```

```
812         //pata 2 esfera 2
813         model = glm::translate(model, glm::vec3(0.0f, 3.0f, 0.0f));
814         model = glm::rotate(model, glm::radians(-90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
815         model = glm::rotate(model, glm::radians(mainWindow.getarticulacionpata2_2()), glm::vec3(0.0f, 0.0f, 1.0f)); // Mover con: P
816
817         modelaux_Pata_Perro_2 = model;//guarda las transformaciones anteriores
818
819         //dibujar una pequeña esfera
820         //no queremos la informacion de la esfera
821         model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
822         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
823         color = glm::vec3(1.0f, 1.0f, 1.0f);
824         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
825         sp.render();
826
827         model = modelaux_Pata_Perro_2;// a partir de la articulacion 2 de la pata 2 (esfera) (ESTE ES MI QUINTO INICIO. 3) <----------------------------------
828
829         //pata 2 cilindro 2
830         model = glm::translate(model, glm::vec3(0.0f, 2.0f, 0.0f));
831
832         modelaux_Pata_Perro_2 = model;//guarda las transformaciones anteriores
833
834         model = glm::scale(model, glm::vec3(1.0f, 2.0f, 1.0f));
835
836         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
837         color = glm::vec3(0.75f, 0.75f, 0.75f);
838         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
839         meshList[2]->RenderMeshGeometry();
840
841         model = modelaux_Cuerpo_Perro;// a partir del centro del prisma rectangular (cuerpo del perro) (ESTE ES MI INICIO) <----------------------------------
```

```
843         //pata 3 esfera 1
844         model = glm::translate(model, glm::vec3(-3.0f, 0.0f, -4.0f));
845         model = glm::rotate(model, glm::radians(-45.0f), glm::vec3(0.0f, 0.0f, 1.0f));
846         model = glm::rotate(model, glm::radians(mainWindow.getarticulacionpata3_1()), glm::vec3(0.0f, 0.0f, 1.0f)); // Mover con: Z
847
848         modelaux_Pata_Perro_3 = model;//guarda las transformaciones anteriores
849
850         //dibujar una pequeña esfera
851         //no queremos la informacion de la esfera
852         model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
853         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
854         color = glm::vec3(1.0f, 1.0f, 1.0f);
855         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
856         sp.render();
857
858         model = modelaux_Pata_Perro_3;// a partir de la articulacion 1 de la pata 3 (esfera) (ESTE ES MI SEXTO INICIO. 1) <---------------------------------
859
860         //pata 3 cilindro 1
861         model = glm::translate(model, glm::vec3(3.0f, 0.0f, 0.0f));
862         model = glm::rotate(model, glm::radians(-90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
863
864         modelaux_Pata_Perro_3 = model;//guarda las transformaciones anteriores
865
866         model = glm::scale(model, glm::vec3(1.0f, 4.0f, 1.0f));
867
868         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
869         color = glm::vec3(0.75f, 0.75f, 0.75f);
870         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
871         meshList[2]->RenderMeshGeometry();
872
873         model = modelaux_Pata_Perro_3;// a partir del centro de la pata 3 (cilindro) (ESTE ES MI SEXTO INICIO. 2) <---------------------------------
```

```
875         //pata 3 esfera 2
876         model = glm::translate(model, glm::vec3(0.0f, 3.0f, 0.0f));
877         model = glm::rotate(model, glm::radians(-90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
878         model = glm::rotate(model, glm::radians(mainWindow.getarticulacionpata3_2()), glm::vec3(0.0f, 0.0f, 1.0f)); // Mover con: X
879
880         modelaux_Pata_Perro_3 = model;//guarda las transformaciones anteriores
881
882         //dibujar una pequeña esfera
883         //no queremos la informacion de la esfera
884         model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
885         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
886         color = glm::vec3(1.0f, 1.0f, 1.0f);
887         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
888         sp.render();
889
890         model = modelaux_Pata_Perro_3;// a partir de la articulacion 2 de la pata 3 (esfera) (ESTE ES MI SEXTO INICIO. 3) <---------------------------------
891
892         //pata 3 cilindro 2
893         model = glm::translate(model, glm::vec3(0.0f, 2.0f, 0.0f));
894
895         modelaux_Pata_Perro_3 = model;//guarda las transformaciones anteriores
896
897         model = glm::scale(model, glm::vec3(1.0f, 2.0f, 1.0f));
898
899         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
900         color = glm::vec3(0.75f, 0.75f, 0.75f);
901         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
902         meshList[2]->RenderMeshGeometry();
903
904         model = modelaux_Cuerpo_Perro;// a partir del centro del prisma rectangular (cuerpo del perro) (ESTE ES MI INICIO) <---------------------------------
```

```
906         //pata 4 esfera 1
907         model = glm::translate(model, glm::vec3(3.0f, 0.0f, -4.0f));
908         model = glm::rotate(model, glm::radians(-45.0f), glm::vec3(0.0f, 0.0f, 1.0f));
909         model = glm::rotate(model, glm::radians(mainWindow.getarticulacionpata4_1()), glm::vec3(0.0f, 0.0f, 1.0f)); // Mover con: Q
910
911         modelaux_Pata_Perro_4 = model;//guarda las transformaciones anteriores
912
913         //dibujar una pequeña esfera
914         //no queremos la informacion de la esfera
915         model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
916         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
917         color = glm::vec3(1.0f, 1.0f, 1.0f);
918         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
919         sp.render();
920
921         model = modelaux_Pata_Perro_4;// a partir de la articulacion 1 de la pata 4 (esfera) (ESTE ES MI SEPTIMO INICIO. 1) <---------------------------------
922
923         //pata 4 cilindro 1
924         model = glm::translate(model, glm::vec3(3.0f, 0.0f, 0.0f));
925         model = glm::rotate(model, glm::radians(-90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
926
927         modelaux_Pata_Perro_4 = model;//guarda las transformaciones anteriores
928
929         model = glm::scale(model, glm::vec3(1.0f, 4.0f, 1.0f));
930
931         glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
932         color = glm::vec3(0.75f, 0.75f, 0.75f);
933         glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
934         meshList[2]->RenderMeshGeometry();
935
936         model = modelaux_Pata_Perro_4;// a partir del centro de la pata 4 (cilindro) (ESTE ES MI SEPTIMO INICIO. 2) <---------------------------------
```

```
//pata 4 esfera 2
model = glm::translate(model, glm::vec3(0.0f, 3.0f, 0.0f));
model = glm::rotate(model, glm::radians(-90.0f), glm::vec3(0.0f, 0.0f, 1.0f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacionpata4_2()), glm::vec3(0.0f, 0.0f, 1.0f)); // Mover con: L

modelaux_Pata_Perro_4 = model;//guarda las transformaciones anteriores

//dibujar una pequeña esfera
//no queremos la informacion de la esfera
model = glm::scale(model, glm::vec3(1.0f, 1.0f, 1.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
color = glm::vec3(1.0f, 1.0f, 1.0f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
sp.render();

model = modelaux_Pata_Perro_4;// a partir de la articulacion 2 de la pata 4 (esfera) (ESTE ES MI SEPTIMO INICIO. 3) <----------------------------------

//pata 4 cilindro 2
model = glm::translate(model, glm::vec3(0.0f, 2.0f, 0.0f));

modelaux_Pata_Perro_4 = model;//guarda las transformaciones anteriores

model = glm::scale(model, glm::vec3(1.0f, 2.0f, 1.0f));

glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
color = glm::vec3(0.75f, 0.75f, 0.75f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el color del objetos
meshList[2]->RenderMeshGeometry();
```

Para hacer que la cola y las cuatro patas con dos articulaciones rotaran realice modificaciones en los archivos window.cpp y window.h.

```
//Para el perro
////////////////////////////////////////////
articulacioncola = 0.0f;
articulacionpata1_1 = 0.0f;
articulacionpata1_2 = 0.0f;
articulacionpata2_1 = 0.0f;
articulacionpata2_2 = 0.0f;
articulacionpata3_1 = 0.0f;
articulacionpata3_2 = 0.0f;
articulacionpata4_1 = 0.0f;
articulacionpata4_2 = 0.0f;
////////////////////////////////////////////
```

```
181     //Para el perro
182     ////////////////////////////////////////////
183     if (key == GLFW_KEY_Y)
184     {
185         theWindow->articulacioncola += 10.0;
186     }
187     if (key == GLFW_KEY_U)
188     {
189         theWindow->articulacionpata1_1 += 10.0;
190     }
191     if (key == GLFW_KEY_I)
192     {
193         theWindow->articulacionpata1_2 += 10.0;
194     }
195     if (key == GLFW_KEY_O)
196     {
197         theWindow->articulacionpata2_1 += 10.0;
198     }
199     if (key == GLFW_KEY_P)
200     {
201         theWindow->articulacionpata2_2 += 10.0;
202     }
203     if (key == GLFW_KEY_Z)
204     {
205         theWindow->articulacionpata3_1 += 10.0;
206     }
207     if (key == GLFW_KEY_X)
208     {
209         theWindow->articulacionpata3_2 += 10.0;
210     }
211     if (key == GLFW_KEY_Q)
212     {
213         theWindow->articulacionpata4_1 += 10.0;
214     }
215     if (key == GLFW_KEY_L)
216     {
217         theWindow->articulacionpata4_2 += 10.0;
218     }
219     ////////////////////////////////////////////
```

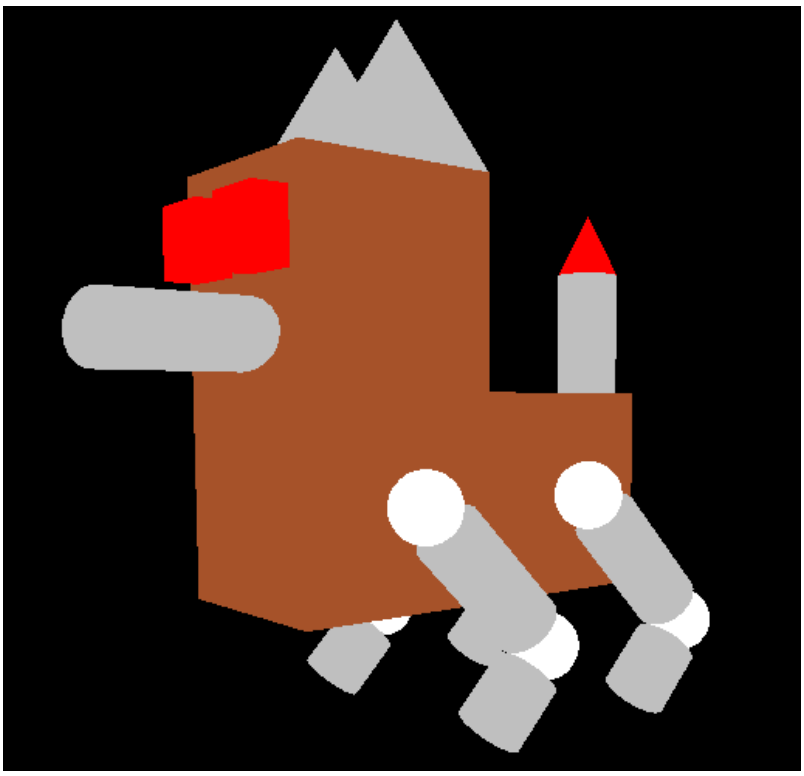
```
34      //Para el perro
35      ////////////////////////////////////////////
36      GLfloat getarticulacioncola() { return articulacioncola; }
37      GLfloat getarticulacionpata1_1() { return articulacionpata1_1; }
38      GLfloat getarticulacionpata1_2() { return articulacionpata1_2; }
39      GLfloat getarticulacionpata2_1() { return articulacionpata2_1; }
40      GLfloat getarticulacionpata2_2() { return articulacionpata2_2; }
41      GLfloat getarticulacionpata3_1() { return articulacionpata3_1; }
42      GLfloat getarticulacionpata3_2() { return articulacionpata3_2; }
43      GLfloat getarticulacionpata4_1() { return articulacionpata4_1; }
44      GLfloat getarticulacionpata4_2() { return articulacionpata4_2; }
45      ////////////////////////////////////////////
```

```
articulacioncola, articulacionpata1_1, articulacionpata1_2, articulacionpata2_1, articulacionpata2_2, articulacionpata3_1, articulacionpata3_2,
```

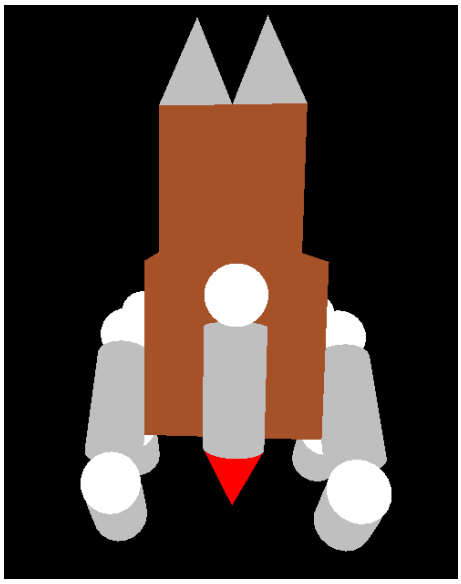
```
articulacionpata4_1, articulacionpata4_2;
```

Ejecución del programa:

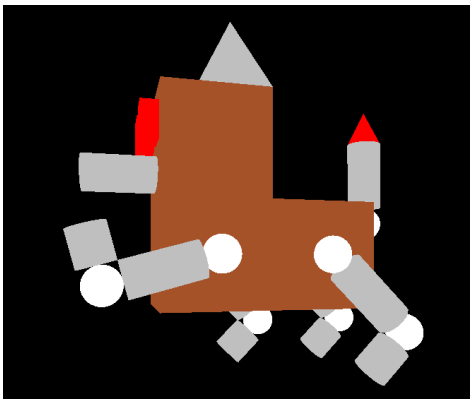
Vista inicial:

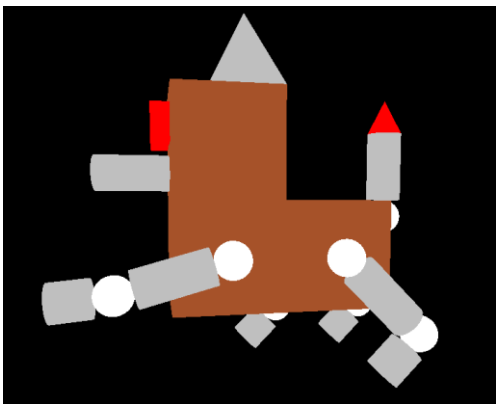


Cola:

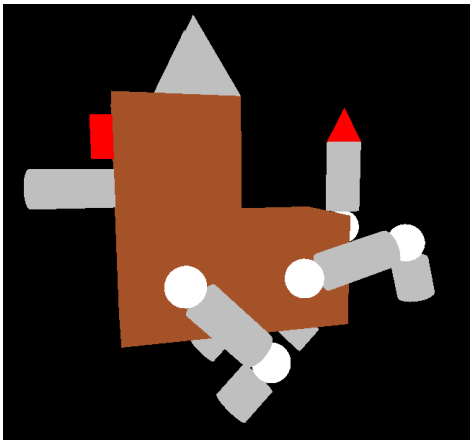


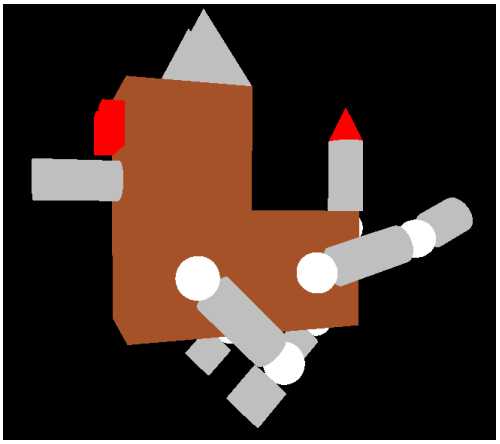
Pata 1 (articulación 1 y 2):



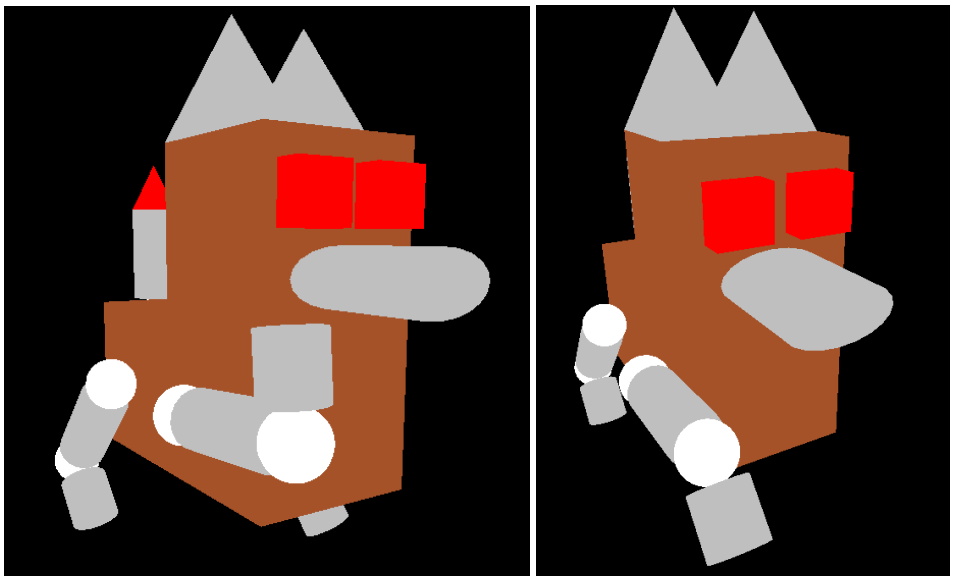


Pata 2 (articulación 1 y 2):

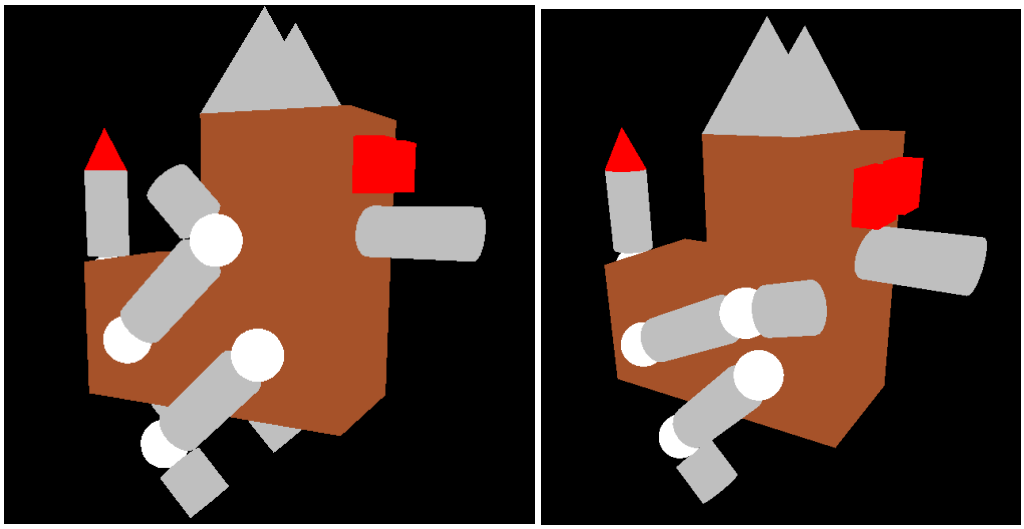




Pata 3 (articulación 1 y 2):



Pata 4 (articulación 1 y 2):



Problemas presentados

1. Matriz auxiliar:
   En un principio me costo trabajo saber cuando y como ocupar las matrices auxiliares, ya que cualquier error podía hacer que la herencia dejar de funcionar de manera correcta.
   "Creo" que al final encontré la forma de usarlas de manera correcta, pues al hacer las pruebas necesarias todo parecía funcionar bien.

Conclusión

A mi parecer, creo que lo que más se me complico en un principio fue como utilizar las matrices auxiliares, ya que si cometía algún error la herencia dejaba de funcionar y las figuras no aparecían en el lugar indicado o no rotaban de manera correcta.

En cuestión de las demás partes del código, comienzo a entender mejor como ocupar los translate, scale y rotate, pues ahora puedo colocar las figuras en las posiciones que quiero.

En general pienso que el objetivo principal de esta práctica se logró, y en cuanto a las matrices auxiliares espero haberlas utilizado de la manera adecuada.

Bibliografía

Studio Minus. (s.f). Colourpicker - Studio Minus. Consultado el 7 de septiembre del 2024 de https://www.studiominus.nl/colourpicker/