Universidad Nacional Autónoma de México

Ciudad Universitaria

Facultad de Ingeniería

Ingeniería en Computación

Laboratorio de Computación Gráfica e Interacción Humano Computadora

Semestre: 2022-1

Alumno: García Lazcano Carlos David

Introducción

Con lo aprendido en clase, se ha podido: texturizar, modelar, reducir y cambiar pivotes según nuestras necesidades con ayuda del software especializado de “Maya”, todo esto para poderlo llevar a OpenGL. Por otro lado, mediante el lenguaje C++, se moldeo los modelos, previamente modificados, para mostrarlos en nuestro ejecutable, así como para realizar los movimientos requeridos por nosotros en nuestra hoja de imágenes de referencia. Así como la asignación de animación para ciertos modelos con ciertas teclas.

Objetivo:

El alumno deberá aplicar y demostrar los conocimientos adquiridos durante todo el curso.

Imagen de referencia:

Interior listado: Televisión, Mesa, Sillas, Gato, Perro, Ratón, Lampara, Pizza y Radio

41804942-4 **Grupo: 08**

Diagrama de Gantt:

	11/10/201	18/10/2021	25/10/2021	01/11/2021	08/11/2021	15/11/2021
Actividad	Semana01	Semana 02	Semana 03	Semana 04	Semana 05	Semana 06
Exportar Modelo de casa						
Exportar modelo gato						
Exportar modelo árbol						
Exportar modelo alfombra						
Exportar modelo piso						
Exportar modelo basura						
Exportar modelo pizza						
Exportar modelo librero						
Exportar modelo mesa						
Exportar modelo buró						
Exportar modelo tv						
Exportar modelo ventilador						
Exportar modelo buró 2						
Exportar modelo lampara						
Exportar modelo sillón						
Exportar modelo silla						
Exportar modelo rata						
Exportar modelo perro						
Exportar modelo radio						
Exportar modelo coche, sin llantas						
Agregar modelos al archivo C++						
Agregar movimiento al ventilador						
Agregar movimiento perro						
Agregar movimiento gato						
Agregar movimiento rata						
Agregar movimiento pájaro						
Agregar movimiento con la tecla a los modelos						
Subir archivo al repositorio						
Crear manual de usuario						

Alcance del proyecto: Recrear un espacio parecido a los modelos de GTA: San Andreas, con 5 animaciones, de las cuales 3 son sencillas y 2 son complejas.

Limitaciones: Si bien no es igual, es parecido, con modelos que sean bajos, en su mayoría; una de las limitantes fue, que al cargar más modelos se tarda más en ejecutar el documento C++, por lo que para modificar cosas mínimas se tarda más en apreciar dichos cambios, por ello ya no se incluyeron más modelos que se pensaron para el modelo.

Documentación:

Para poder crear el archivo se utilizó el archivo de la práctica número 5, en este, se agregaron los modelos, variables de estados para el movimiento de los modelos de; perro, gato, rata, ventilador y un pájaro.

Imagen correspondiente para la carga de modelos.

```
// Carga de modelos
Model casa((char*)"Models/casa/casa.obj");
Model piso((char*)"Models/piso/piso.obj");
Model gato((char*)"Models/cat/gato.obj");
Model arbol((char*)"Models/arbol/arbol.obj");
Model basura((char*)"Models/basura/basura.obj");
Model mesa((char*)"Models/mesa/mesa.obj");
Model pizzza((char*)"Models/pizzza/pizzza.obj");
Model Librero((char*)"Models/librero/librero.obj");
Model carpet((char*)"Models/alfo/carpet.obj");
Model pasto((char*)"Models/pasto/pasto.obj");
Model buro((char*)"Models/buro/buro.obj");
Model tv((char*)"Models/tv/tv.obj");
Model fan((char*)"Models/fan/fan.obj");
Model beu((char*)"Models/beu/beu.obj");
Model lampa((char*)"Models/lamp/lampa.obj");
Model coach((char*)"Models/coach/coach.obj");
Model dog((char*)"Models/dog/dogo.obj");
Model rat((char*)"Models/rat/rat.obj");
Model van((char*)"Models/van/van.obj");
Model bird((char*)"Models/bird/bird.obj");
Model radio((char*)"Models/radio/radio.obj");
Model silla((char*)"Models/silla/silla.obj");
```

Imagen correspondiente de la animación del ventilador:

```
//VENTILADOR
float rotfan= 0.0f;
bool fanrot;
```

Imagen correspondiente de la animación, movimientos y recorridos del perro:

```
//PERRO
bool dogmov;//activar a
float rotdog = 0.0f;//r
float rotdog2 = 0.0f;//
float dogmovx = 0.0f;//
float dogmovy = 0.0f;//
//recorrido perro
bool recodog1 = true;
bool recodog2 = false;/
bool recodog3 = false;/
bool recodog4 = false;
bool recodog5 = false;/
bool recodog6 = false;/
bool recodog7 = false;/
bool recodog8 = false;
```

Imagen correspondiente de la animación, movimientos y recorridos de la rata:

```
//RATA
bool ratmov;//moviminet
float rotrat = 315.0f;/
float rotrat2 = 0.0f;//
float movratx = 0.0f;
float movraty = 0.0f;//
float movratz = 0.0f;
//recorrido rata es cic
bool recorat1 = true;//
bool recorat2 = false;/
bool recorat3 = false;/
bool recorat4 = false;/
bool recorat5 = false;/
bool recorat6 = false;/
```

Imagen correspondiente de la animación, movimientos y recorridos del pájaro:

```
//PAJARO
bool movpaja;//animac
float rotpaja = 0.0f;
float movpajx = 0.0f;
float movpajz = 0.0f;
//recorrido pajaro
bool recopaj1=true;
bool recopaj2=false;/
bool recopaj3=false;
bool recopaj4=false;/
```

Imagen correspondiente de la animación, movimientos y estados del gato:

```
//GATO
bool catmov;//para que s
float movcatx = 0.0f;//m
float movcatz = 0.0f;//m
float rotcat = 90.0f;//r
//recorridos gato son ci
bool recocat1 = true;//s
bool recocat2 = false;
bool recocat3 = false;
bool recocat4 = false;
```

Si bien la utilización de las variables para modificar la posición es sencilla, es necesario registrar y explicar dichas variables:

Perro:

En la siguiente imagen se muestra la posición y rotación del modelo, rotdog hace que el perro se incline para simular un salto, y rotdog2 se encarga de voltear al perro, dependiendo de la dirección donde corra; dogmovx lo desplaza en eje “x”, y dogmovy desplaza a nuestro modelo en el eje “y”. Cuenta como animación compleja.

```
//PERRO
model = glm::mat4(1);
model = glm::translate(model, glm::vec3(18.0f+dogmovx, -0.32f+dogmovy, 12.0f));//recorrido del perro
model = glm::rotate(model, glm::radians(rotdog), glm::vec3(0.0f, 0.0f, 1.0f));//rotación que simula el
model = glm::rotate(model, glm::radians(rotdog2), glm::vec3(0.0f, 1.0f, 0.0f));//rotación que simula el
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
dog.Draw(shader);
```

Gato:

En la imagen se muestra que rotcat se utiliza para rotar al gato dependiendo del estado de su movimiento en el que se encuentra, mientras que movcatx mueve al modelo en el je x, y movcatz lo mueve en el eje z. Cuenta como movimiento sencillo.

```
model = glm::mat4 (1);
model = glm::scale(model, glm::vec3(0.5f, 0.5f, 0.5f));//gato algo grande, se opto por reducirlo aquí
model = glm::translate(model, glm::vec3(0.3f + movcatx, 1.95f, -3.0f+ movcatz));//posisción original má
model = glm::rotate(model, glm::radians(rotcat), glm::vec3(0.0f, 1.0f, 0.0f));//cse rota cuando se nece
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
gato.Draw(shader);
```

Rata:

Al igual que con el modelo del perro, se desplazará el modelo en el eje “y”, pero con este modelo se desplazará en los 3 ejes; al igual que el perro, un movimiento lo hará en 2 ejes x & z, mientras que para el desplazamiento en ele “y”, será lineal, dependiendo del movimiento o dirección que realizará la rata a nivel del suelo, para eso se usará rotrat; mientras que rotrat2 se usará para cuando suba por la pata de la mesa. Esta cuenta como movimiento complejo.

```
//RATA
model = glm::mat4(1);
model = glm::scale(model, glm::vec3(0.75f, 0.75f, 0.75f));//muy grande, se redimensiono para ser más es
model = glm::translate(model, glm::vec3(3.5f + movratx, 0.05f + movraty, -1.5f + movratz));//simula el
model = glm::rotate(model, glm::radians(rotrat), glm::vec3(0.0f, 1.0f, 0.0f));//orientación de la direc
model = glm::rotate(model, glm::radians(rotrat2), glm::vec3(1.0f, 0.0f, 0.0f));// orientación al escala
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
rat.Draw(shader);
```

Pájaro:

Este modelo solo se desplaza en los ejes “x” & “z”, se encuentra a una altura apenas sobre el techo, sus respectivas variables movpajx y movpajz le dan su movimiento, mientras que rotpaja le da la dirección para que sea acorde al vuelo que realiza. Animación sencilla.

```
//PAJARO
model = glm::mat4(1);
model = glm::translate(model, glm::vec3(-12.0f + movpajx, 8.0f, -12.0f + movpajz));// animación del vue
model = glm::rotate(model, glm::radians(rotpaja), glm::vec3(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
bird.Draw(shader);
```

Ventilador:

Solo rota simulando su comportamiento común. Animación sencilla.

```
//ventilador
model = glm::mat4(1);
model = glm::translate(model, glm::vec3(0.0f, 3.9f, 0.0f));
model = glm::rotate(model, glm::radians(-rotfan), glm::vec3(0.0f, 1.0f, 0.0f));//gira como uno normal,
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
fan.Draw(shader);
```

Movimiento cámara:

Para mover la cámara es necesario presionar las teclas A o ← para ir a la izquierda, para ir a la derecha con la tecla D o →, para acercarte es la tecla W o ↑, y finalmente para alejarse es la tecla S o ↓.
Para implementar esta parte se necesito de las funciones DoMovement, donde dependiendo de las teclas presionadas se hace el movimiento de la cámara.

Animaciones:

Para el uso de las animaciones se uso la función DoMovement, es este apartado se colocaron cada una de las 5 animaciones, todas ellas dependiendo de una variable booleana, que se vuelve verdadero o falsa en la función KeyCallBack.
Debido a que esta parte es sumamente larga, en los diagramas de estados/recorridos de los objetos se muestra de donde parte y hasta dónde llegan dichos recorridos de su respectivo objeto.

Para KeyCallBack se utilizaron las variables booleanas que se consultaran en DoMovement, aquí cada booleano contiene el nombre más corto posible, ya sean en inglés o español, más el prefijo mov, para saber cuál es de qué objeto.

```
if (keys[GLFW_KEY_Q]) {/
    fanrot=!fanrot;
}
if (keys[GLFW_KEY_E]) {/
    catmov = !catmov;
}
if (keys[GLFW_KEY_R]) {/
    ratmov = !ratmov;
}
if (keys[GLFW_KEY_P]) {/
    dogmov = !dogmov;
}
if (keys[GLFW_KEY_B]) {/
    movpaja = !movpaja;
}
```

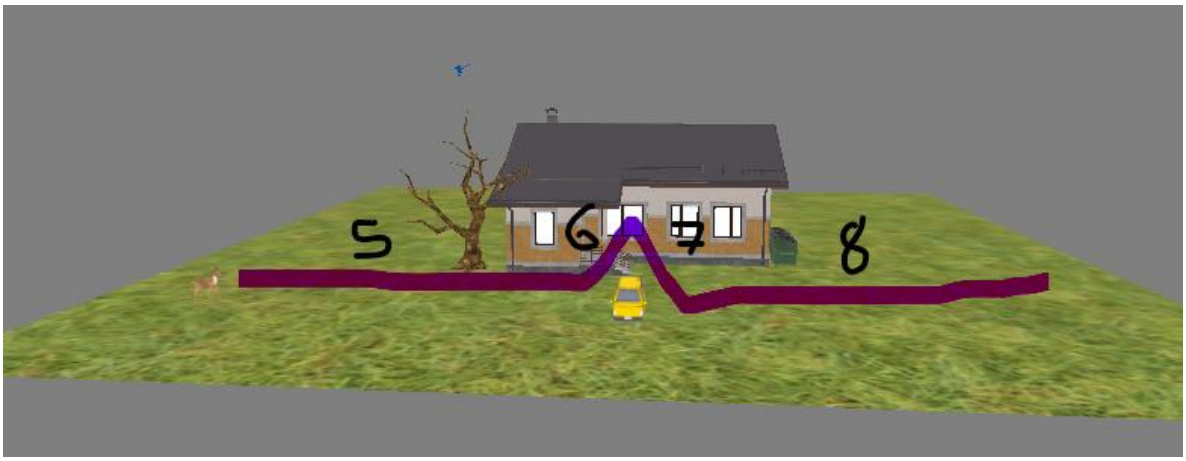
Ventilador:

Como es sencillo, no cuenta con estados, por ello no se mostrará su animación, solo su código, su animación se activa con la tecla **Q**.

```
//MOVIMINETO VENTILADOR
  if (fanrot) {
      rotfan += 0.1f;
  }
  else if (!fanrot) {
      rotfan += 0.0f;
  }
```

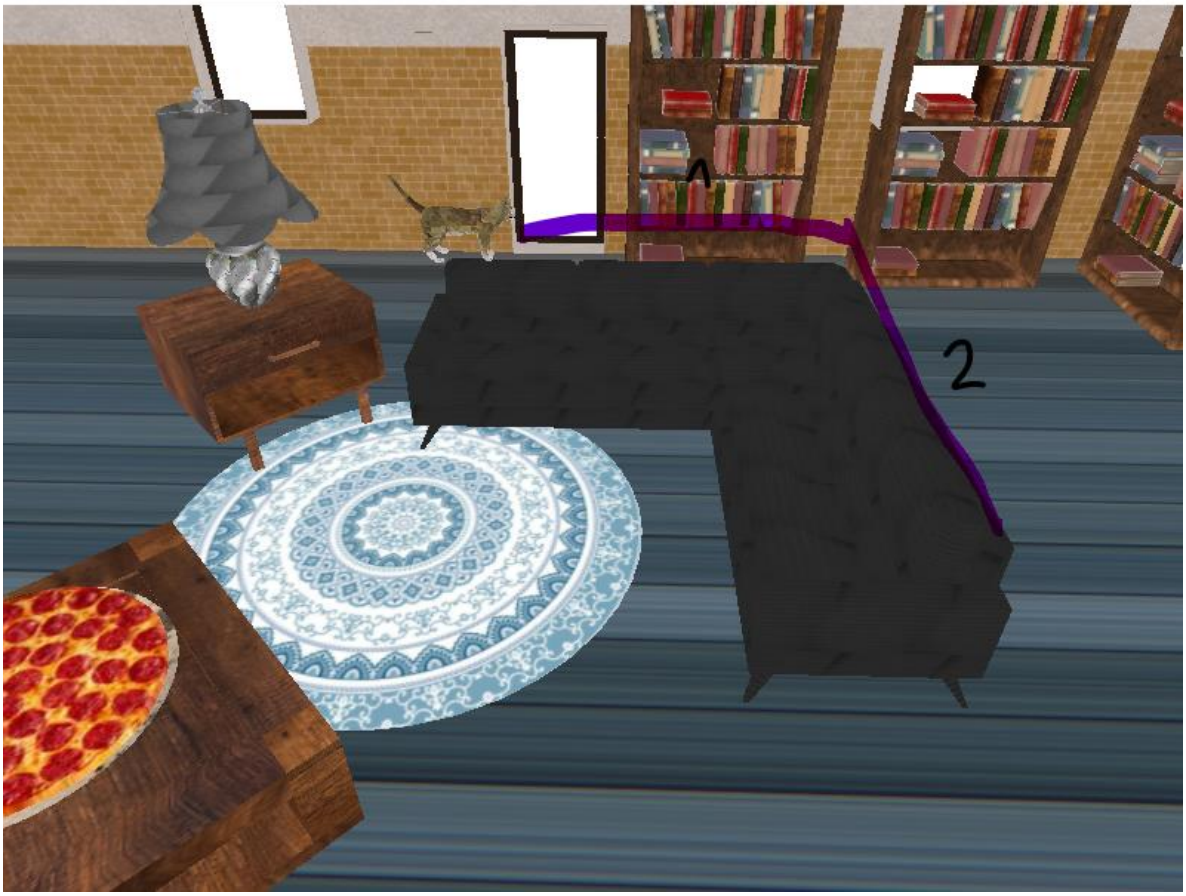
Perro:

Para la activación de está animación solo se necesita presionar la tecla **P**, y para detenerla se necesita presionarla de nuevo, realiza su recorrido, pero solo se puede activar o detener. Cuando llega al estado 8, su estado siguiente será el 1, por lo tanto, su animación es cíclica.



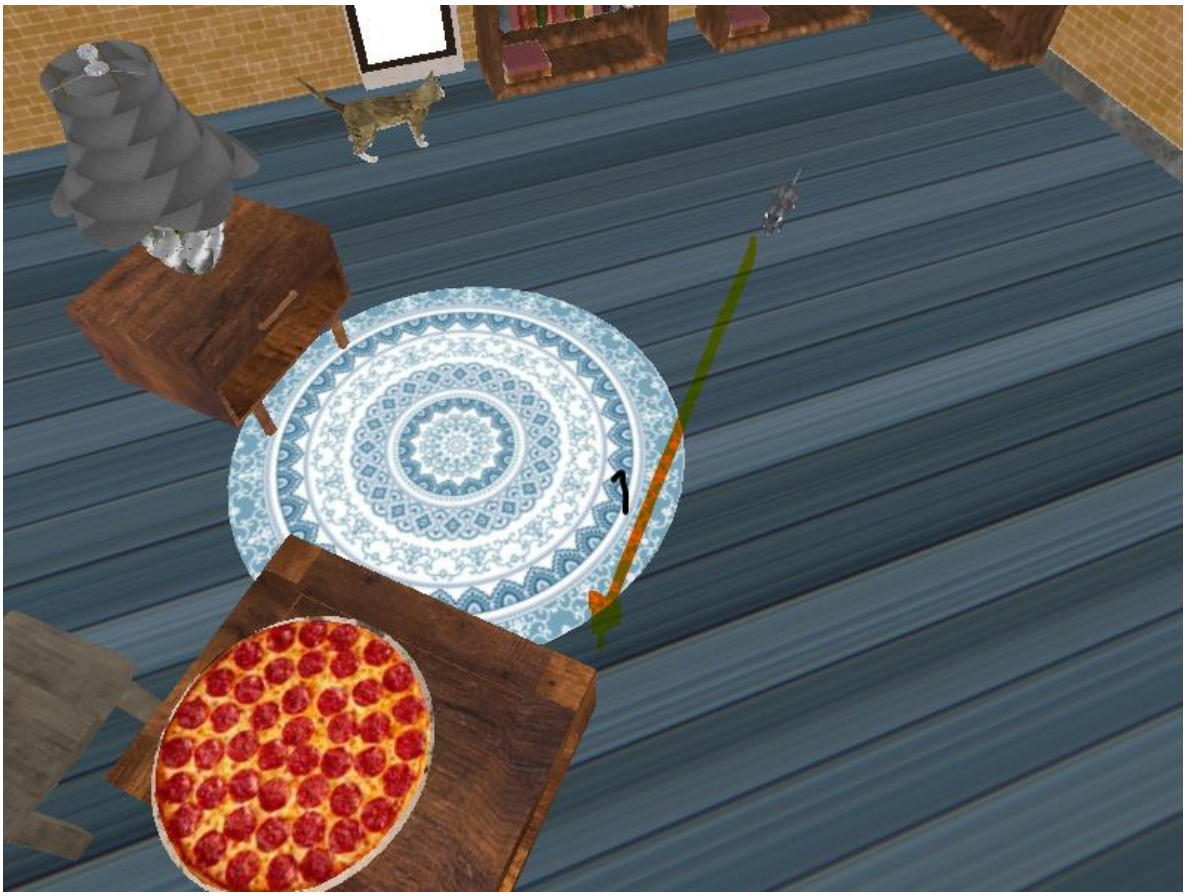
Gato:

La animación de este modelo es en forma de L, como se muestra en sus siguientes movimientos, cuando se llega a al movimiento 4, vuelve a hacer el primero, por lo tanto, es cíclico. Para la activación de esta animación se necesita presionar la tecla **E**:


1
2

Rata:

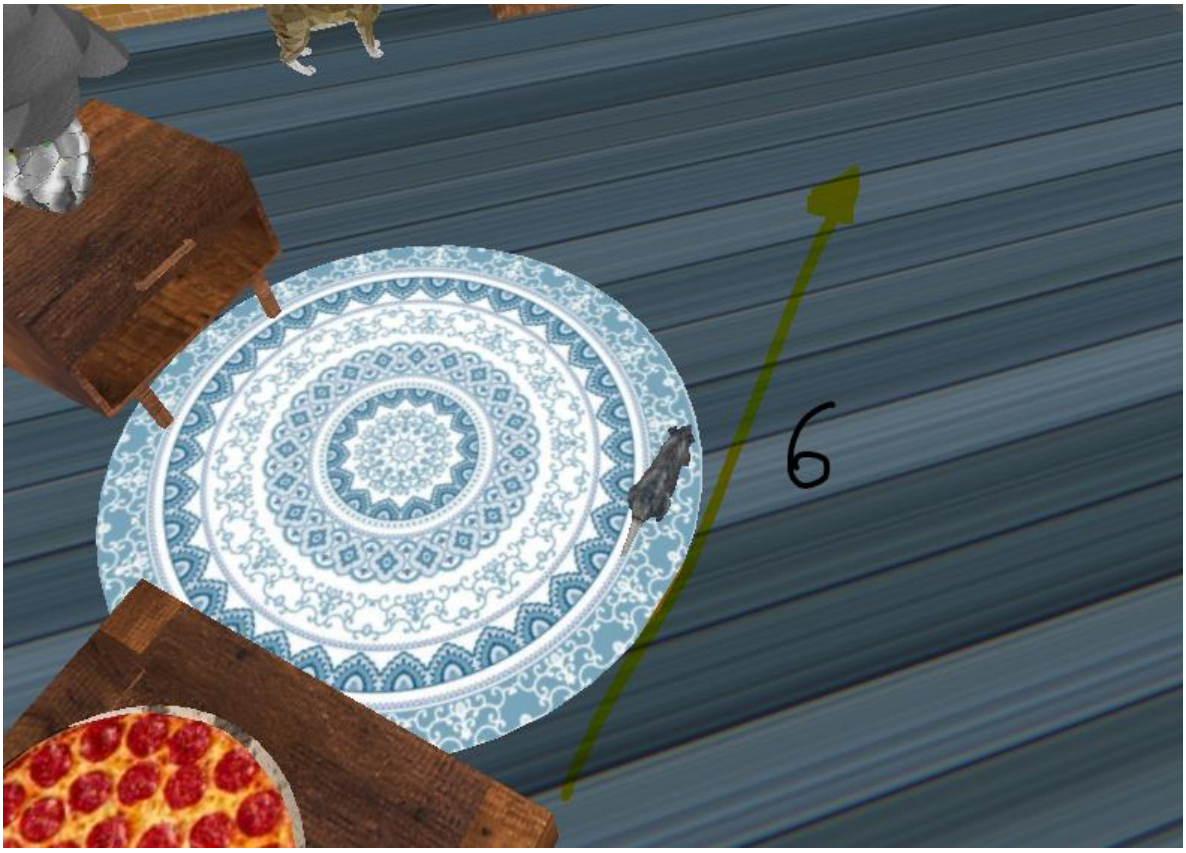
Para apreciar mejor el movimiento de la rata, fue necesario quitar el sillón. La animación es cíclica, se activa con la tecla **R**.


1


2

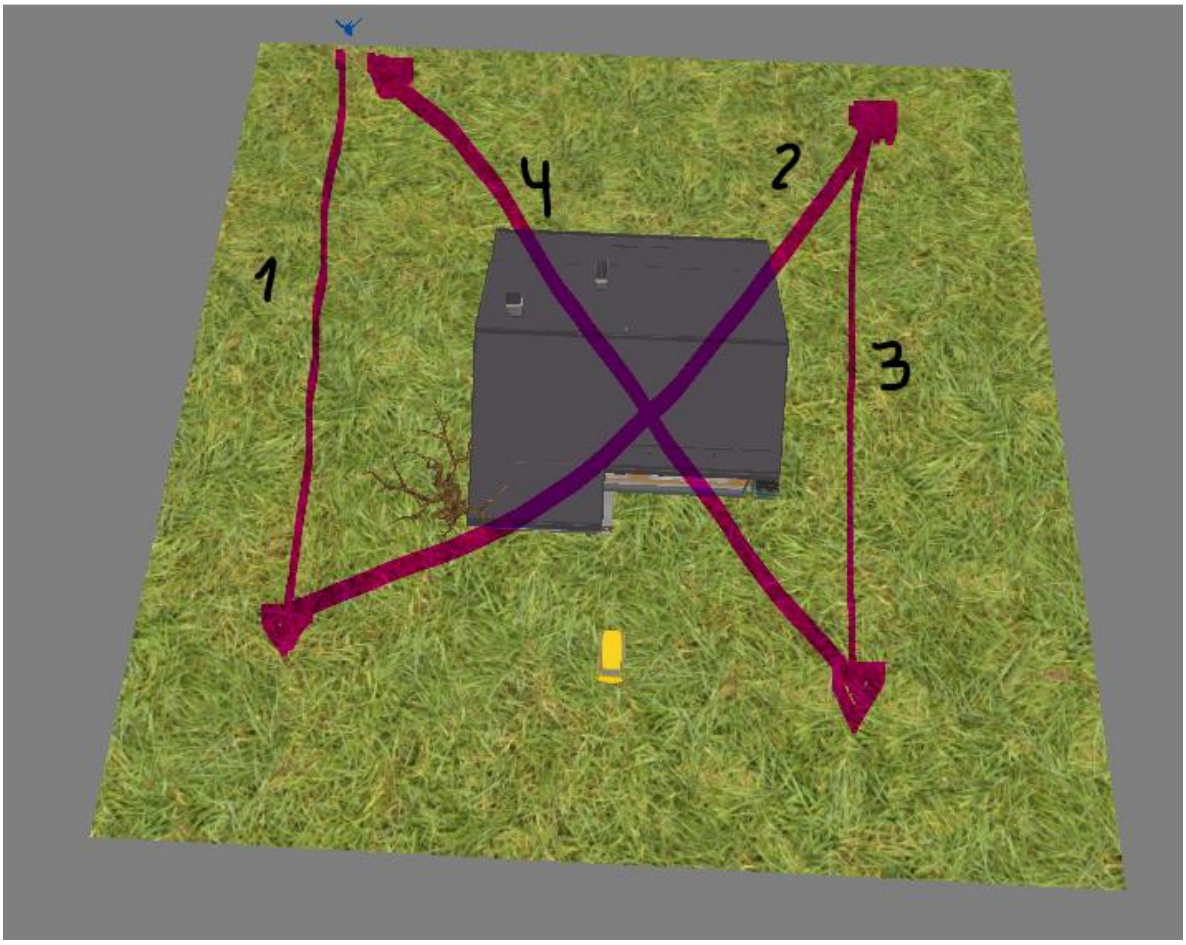

3


4



Pájaro:

Al igual que el resto de las animaciones, solo se puede activar y detener, en este caso será con la tecla **B**



Conclusiones:

Los objetivos se lograron, puesto que se implementaron la mayoría de los conceptos aprendidos durante el curso, por lo que estoy satisfecho con el curso y la realización de dicho proyecto.

Link: https://github.com/CarlosLazcano/41804942-4_ProyectoFinal_GPO8