Universidad de Chile
Facultad de Ciencias Físicas y Matemáticas
Departamento de Ciencias de la Computación

Tarea 2

OpenGL 3D

Integrantes: Esteban Zúñiga S.
Profesor: Nancy Hitschfeld K.
Auxiliar: Andrés Cerda
Pablo Pizarro R.
Alonso Utreras
Ayudantes: Tomás Calderón R.
Beatriz Graboloza M.
Sebastián Olmos H.
Nadia Decar
Ayudante de laboratorio: Sergio Alvarez

Fecha de entrega: 9 de diciembre de 2020
Santiago, Chile

1. Solución propuesta

Para la realización de esta tarea, al igual que en la tarea 1 se escogió desarrollar el clásico juego "Snake", pero esta vez en 3D.

El desarrollo de la tarea consistió en la realización de dos scripts, "modelo" y "vista".

El primero, contiene las funciones para crear la serpiente, la zanahoria y las murallas. Para las manzanas se utilizaron dos círculos, luego fueron modeladas utilizando modelación jerárquica, Para la creación de la serpiente, se utilizó modelación jerárquica, Se diseñó una función que creaba una serpiente como un cubo con textura, luego otra función que se encargaba de dibujar la serpiente de largo N, siendo $N-1$ las zanahorias que se ha comido la serpiente.

En el script "vista" posee todo lo que ocurre en pantalla, como es el dibujo de la serpiente, las zanahorias y el escenario de juego. También evalúa una posible colisión, ya sea con la muralla o consigo misma, en este caso se despliega el mensaje de "Game Over", etc. Algunas funcionalidades importantes que se heredaron de la tarea 1 son (las funcionalidades tienen su correspondiente explicación):

· ¿Cómo crece la serpiente?: La serpiente, a medida que come una zanahoria, crece su tamaño en una unidad, esta unidad corresponde a un cubo de arista 1. La forma en que crece es simple, se agrega una unidad extra en la cola de la serpiente, posición que se encuentra guardada en una variable.

· ¿Cómo se mueve la serpiente?: Luego de observar que mover todos los cubos que componen la serpiente no era muy eficiente, se encontró una forma de hacer este proceso más eficiente, el cual consiste en mover la cola de la serpiente delante de la cabeza, así, solo se traslada solo un cubo por movimiento. Esto se logra accediendo y trasladando al último nodo de la serpiente creada, luego al penúltimo y así sucesivamente, cuando se ya se han trasladado todos los nodos, se recorre nuevamente desde el último. El siguiente dibujo ilustra de mejor manera el proceso:

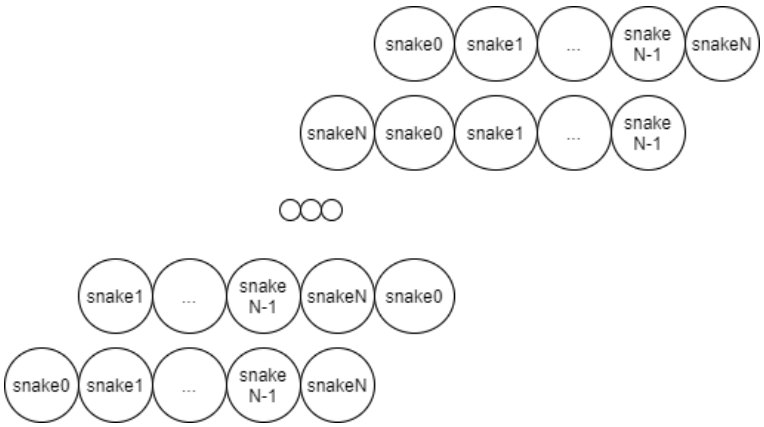

Figura 1: Movimiento de serpiente

¿Qué ocurre cuando el jugador pierde?: En el momento que la serpiente choca contra algún borde, o contra ella misma, el jugador pierde, desaparecen las figuras y se despliega una textura animada con un giro parcial con la frase "GAME OVER".

Las funcionalidades anteriormente mencionadas se aplicaron de igual forma que en la tarea 1. La siguiente funcionalidad tuvo una variación.

· Función update: Se creó una función update, que dependiendo de la variable *state*, indica hacia donde se debe mover la serpiente, por ejemplo si $state == "RIGHT"$, la serpiente se moverá hacia la derecha. La función update se llama cada i ciclos *while*, siendo $i = 30 - (5 * largoSerpiente)//8$. Se escogió este valor para i debido a que si se escoge solo 30, la serpiente se avanza más lento, esto debido a que mientras más grande la serpiente, el ciclo while demora más en ejecutarse. También, dependiendo de la variable *state*, se modificaba el valor de x e y desde donde miraba la cámara, logrando que la cámara siempre esté tras la serpiente y mirando hacia adelante.

· Cámara: En el juego hay tres cámaras distintas, las cuales son las siguientes.

. Cámara principal: Esta cámara siempre mira a la serpiente desde atrás, logrando un efecto de seguimiento sobre la cabeza de dicho personaje. Para implementarla se crearon las variables x_cam e y_cam, las cuales cambian con cada llamada a la función update. Además, cuando el jugador hacía virar la serpiente, la cámara también viraba, posicionándose a la misma distancia de siempre de la cabeza, pero desde un distinto ángulo. En esta cámara solo se utilizan las teclas A (o flecha izquierda) y D (o flecha derecha). Más adelante se explicarán los controles del juego.

. Cámara que apunta desde arriba hacia abajo: Esta cámara apunta desde una altura en el eje z tal que se vea el escenario completo. Con esta cámara se puede ocupar $WASD$ y todas las flechas.

. Cámara con vista en perspectiva: Esta cámara ve en perspectiva desde una arista del mapa, esta cámara apunta hacia el centro del escenario con cierta altura en el eje z tal que se vea todo el escenario, pero con un desvío en el eje y. Con esta cámara se puede ocupar $WASD$ y todas las flechas.

2. Instrucciones de ejecución

Para ejecutar el código debe dirigirse al directorio *tarea2c* y escribir el comando que aparece a continuación:

```
python vista.py
```

3. Controles

Primero, notar que el juego comienza con la cámara principal. Se tienen dos opciones de cambio. Para cambiar a la cámara que apunta de arriba hacia abajo se presiona la tecla E. Para cambiar a la cámara con vista en perspectiva se debe presionar la tecla T. Presionando la tecla R se puede volver a la cámara principal.

El control de la serpiente depende de la cámara que se ocupa.

Con la cámara principal, la serpiente se puede controlar con las teclas AD o las flechas izquierda y derecha de su computador, las teclas WS y las flechas arriba y abajo no se ocupan, ya que la cámara acompaña a la serpiente desde atrás, dejando solo la posibilidad de moverse hacia los lados.

Con las otras dos cámaras se activa la opción de moverse con $WASD$ y todas las flechas.

Para efectos de revisión, se activó la posibilidad de ver las figuras sin relleno, presionando *espacio*.

4. Resultados

A continuación, se adjuntan algunas imágenes del juego en funcionamiento:

Figura 2: Cámara principal

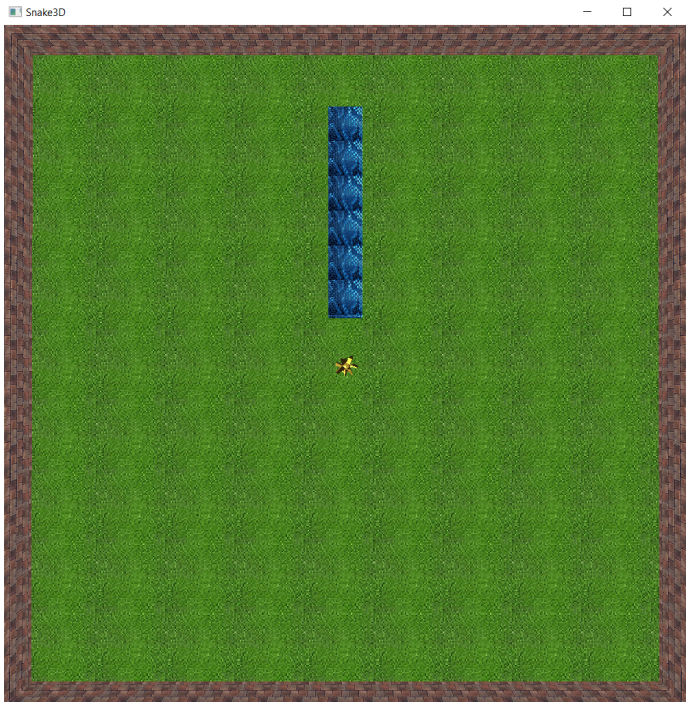

Figura 3: Cámara que apunta de arriba a abajo

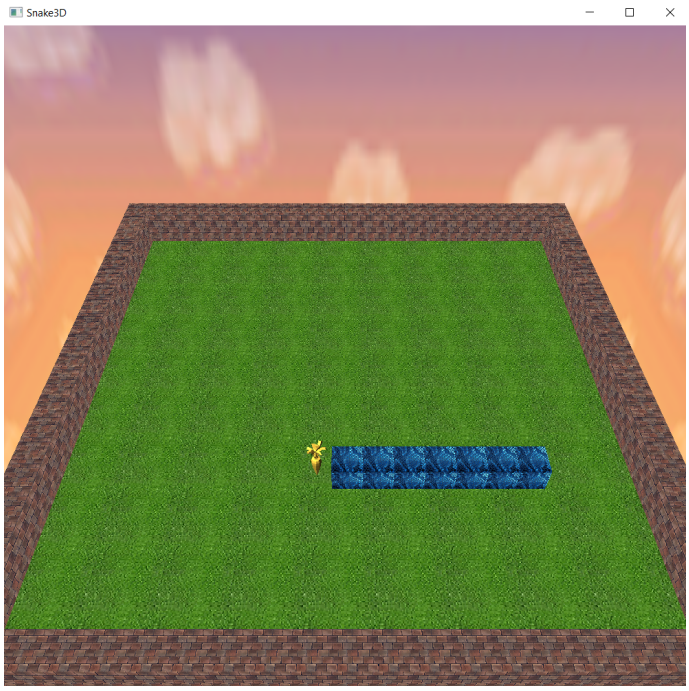

Figura 4: Cámara en perspectiva

Figura 5: GAME OVER

*Link al repositorio en GitHub del proyecto: https://github.com/estebanzuniga/my_snake
**En caso de encontrar un bug, por favor avisar.*