Reporte Tarea 2A

Tomás Maturana

Solución Propuesta

La solución propuesta no cumple con todos los requisitos presentes en el enunciado. Lo logrado fue:

- Crear mallas poligonales para el cielo y suelo de la cueva a partir de una matriz en formato .npy. Ambas matrices están modeladas con funciones matemáticas en el archivo mesh_creator.py, luego en cave.py se ocupan para crear las mallas. Ambas cuevas no tienen un patrón simple, para simular una cueva irregular. Tanto el cielo como el suelo de cada cueva, tienen en sus caras tres índices de texturas al azar (3 para el cielo y 3 distintas para el suelo)
- El modelo de Lara es un cubo alargado en el eje z, que tiene en sus caras laterales la textura de una persona por atrás. Al avanzar o retroceder, la textura que se muestra cambia para dar la impresión de que el personaje está caminando. Cabe destacar que Lara cabe contenida en una cara de la malla. Además, existe la clase Player, que es donde se implementa la lógica de lo que Lara debe hacer y donde se encuentra referenciado el modelo. Si bien en el enunciado se pide que bastaba con una animación 2D para el movimiento de Lara, mi plan fue hacerle todas las vistas posibles, armando un modelo 3D, pero por tiempo no alcancé.
- No se alcanzó a implementar el choque entre piezas del puzzle y Lara. Si bien hay código que sirve para implementarlo posteriormente en la tarea recuperativa, actualmente la solución sólo despliega piezas al azar por la cueva, los cuales giran para poder verlos a mayor distancia.
- Se tiene la clase Controller, con atributos que sirven para mover la cámara y a Lara al captar las señales de teclado y mouse, para mover la linterna, para determinar la luminosidad de la linterna, y para mantener una referencia a la cámara.
- La cámara setea sus valores iniciales de at y eye según el archivo map.npy que se le entregue, ya que ambos tienen distintas posiciones accesibles. Luego, en cada iteración del programa, se recalcula el valor del at y del eye según la dirección en la que Lara cambia de posición o la dirección de la vista.
- La posición de Lara en la pantalla está determinada por el vector at de la cámara, el cual modifica sus valores de X e Y según cómo se mueve Lara, y el valor de Z es calculado a partir de la malla del suelo y según el valor de X e Y de ese momento. De esta manera, Lara se va moviendo sobre la superficie del suelo. Lamentablemente esto no sucede siempre; hay momentos en los que Lara queda flotando, o bajo tierra, esto se debe a que la posición depende del at de la cámara, y éste no se logró calcular perfectamente, a pesar de todas las horas dedicadas a calibrarlo.
- Se logró que Lara "choque" contra la cueva, en caso de que la altura del cielo menos la del suelo, fuera menos de 1. Para comprobar esta sección, recomiendo usar el mapa 1 (map1.npy), ya que hay secciones donde uno puede "mirar" a través de espacios que quedan entre el suelo y el cielo, pero se puede cruzar sólo por los que son suficientemente grandes.
- También se logró que no se puedan acceder a sectores que tienen una pendiente *muy grande*, particularmente, si la diferencia de altura entre donde está y hacia donde se desea mover Lara, es mayor a 2, entonces tiene una pendiente muy grande para avanzar. Esto difiere del enunciado respecto a que se pedía que fueran pendientes de 45° a las que Lara no podría acceder, pero no se logró tal cual. Recomiendo el mapa 2 (map2.npy) para comprobarlo).

- La linterna se logró parcialmente, ya que quedó media bizca; al parecer porque al girar la cámara, el eye no se modifica, y como la dirección de la lintema se calculó a partir del at y el eye, queda apuntando a lugares que no debería (Apunta al 0,0,0). Aún así se arregló parcialmente al calcularle ciertos ajustes; por lo tanto cumple su función de linterna. Tiene 3 niveles de luminosidad, siendo 1 casi apagado, y 3 casi con iluminación total.

El grafo de escena es bastante simple, ya que sólo hay un nodo que contiene a todos los demás nodos: fullScene tiene de hijos a playerNode, caveScene, y a cada pieza del puzzle. Cabe destacar que tanto playerNode como los nodos de las piezas del puzzle, tienen un nodo específico para las transformaciones entre medio del nodo fullScene y el shape final (cubo con textura).

Instrucciones de Ejecución

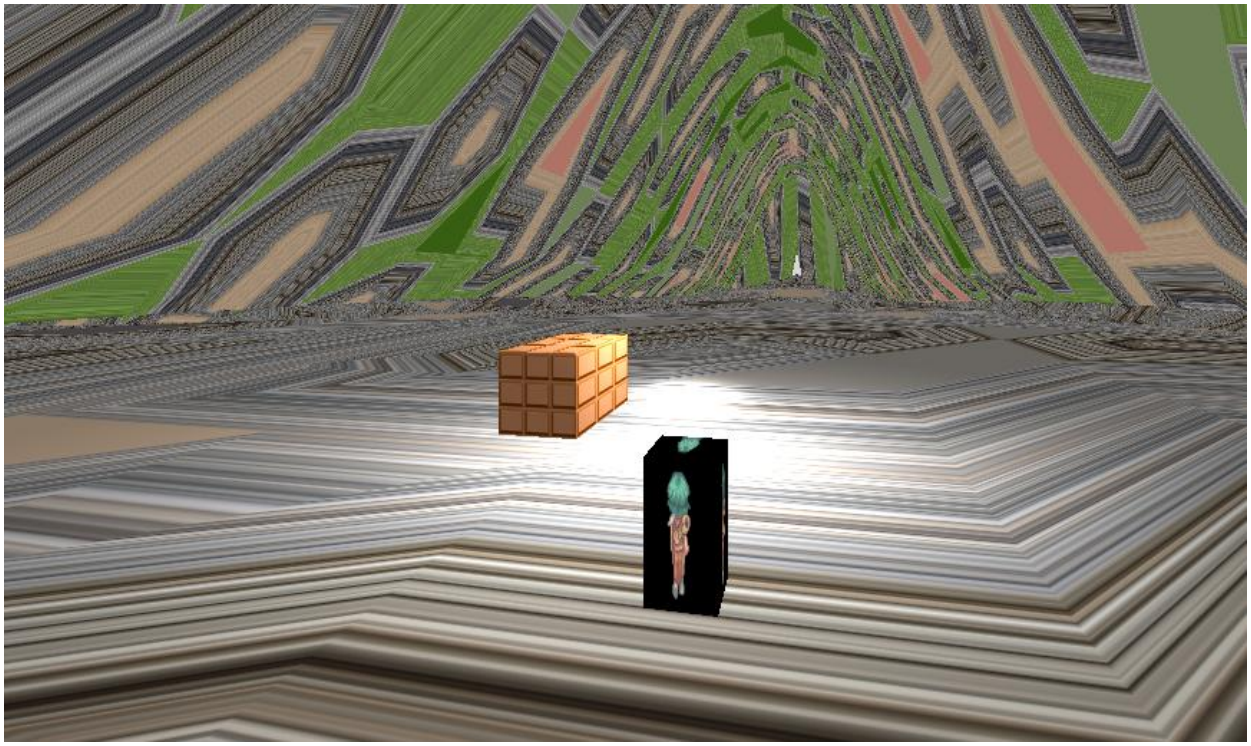
Teniendo las librerías del curso instaladas y estando en la carpeta donde se encuentra cave.py y sus archivos relacionados, se ejecuta el archivo con la sintaxis "*cave.py **map** textures.png **N**",* siendo N el número de piezas del puzle que se desean tener, y map la referencia al archivo que contiene la matriz para crear la malla. Hay dos opciones de map: **map1.py** y **map2.py**.

Una vez entra en ejecución el programa, Lara aparecerá en la zona central de la ventana, y se podrá mover la cámara en torno ella según la posición del mouse, siendo más brusco el movimiento si el mouse está más lejos del centro. Haciendo click derecho, Lara se moverá hacia atrás, y haciendo click izquierdo hacia el frente. También es posible mover a Lara con las flechas del teclado. Si se desea configurar la linterna, se podrá elegir entre presionar el 1, el 2 o el 3 para ajustar su intensidad. Además, se puede presionar el 0 para apagar la linterna por completo. Para mejorar un poco más la precisión y funcionalidad de la linterna, se agregó la posibilidad de cambiar su dirección apretando las teclas W, A, S, D.

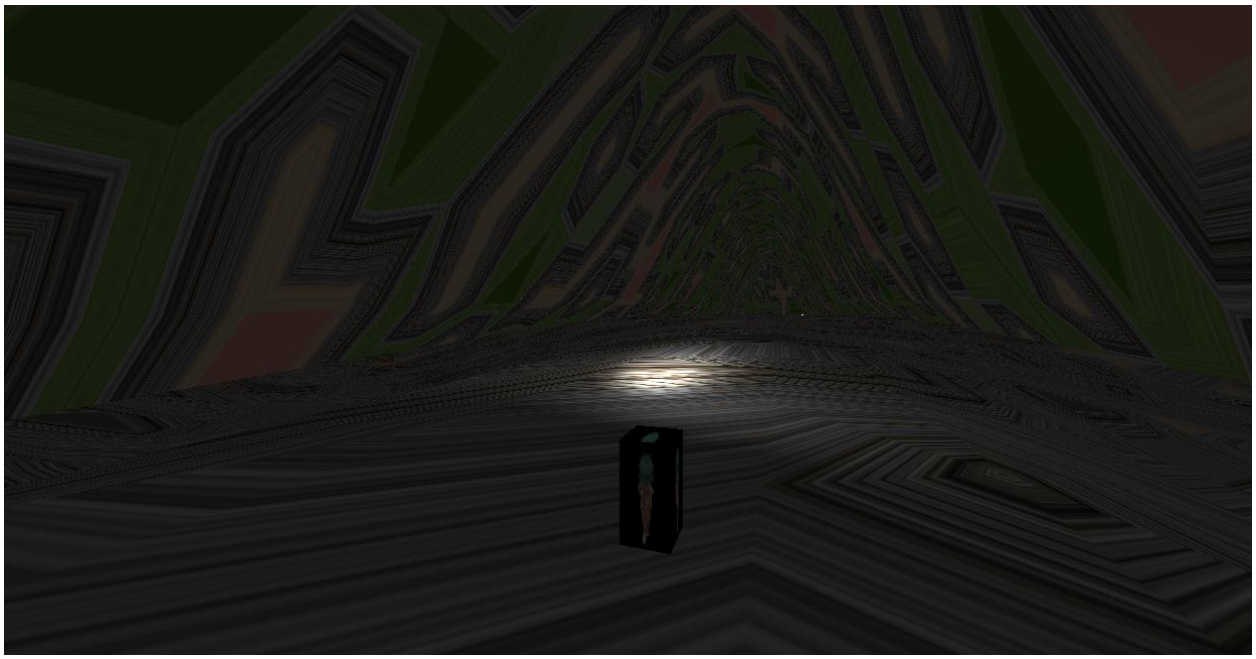
Lamentablemente no se pudo terminar la finalidad del juego, ya que no están implementados los choques con las piezas del puzzle, por lo tanto tampoco se puede "ganar".

Resultados

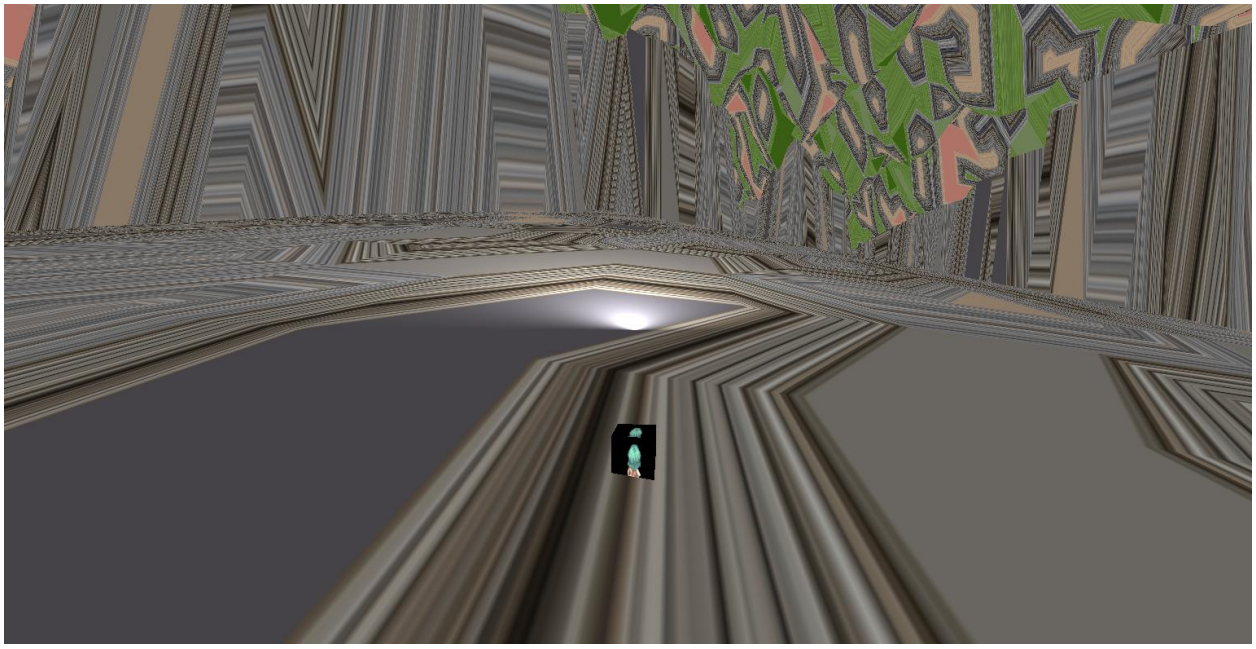
Lara y una pieza del puzzle en el mapa 1 con linterna con luminosidad 3 (máxima):



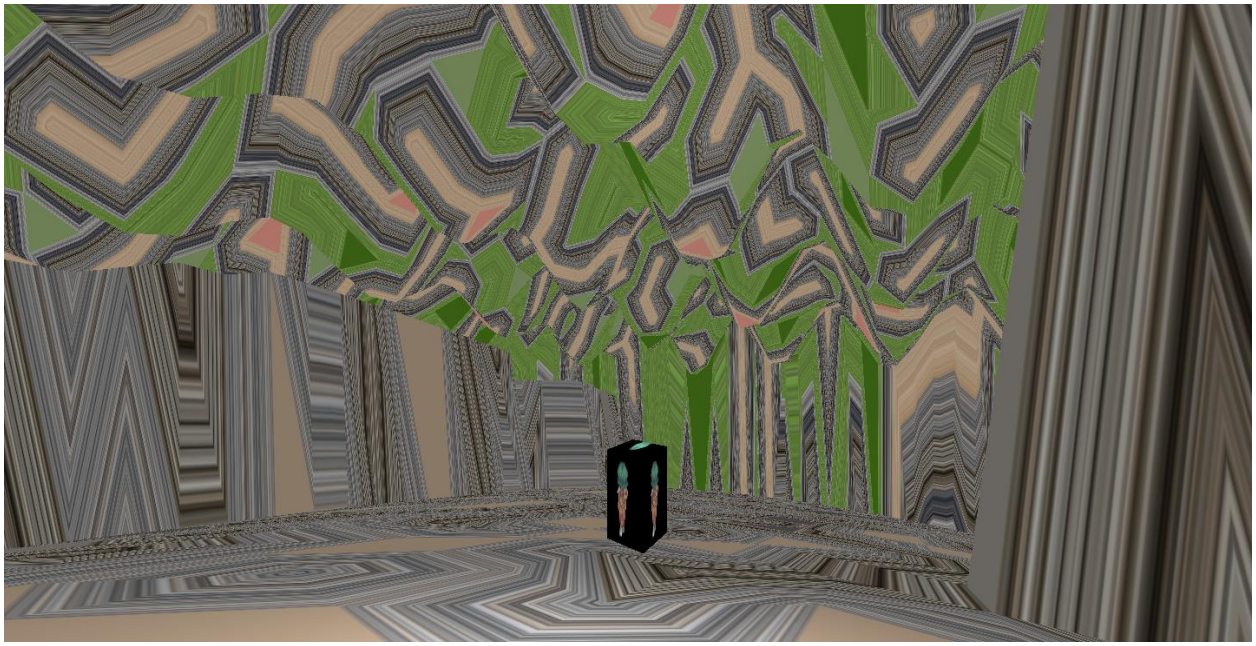
Linterna con luminosidad 2 apuntando al frente



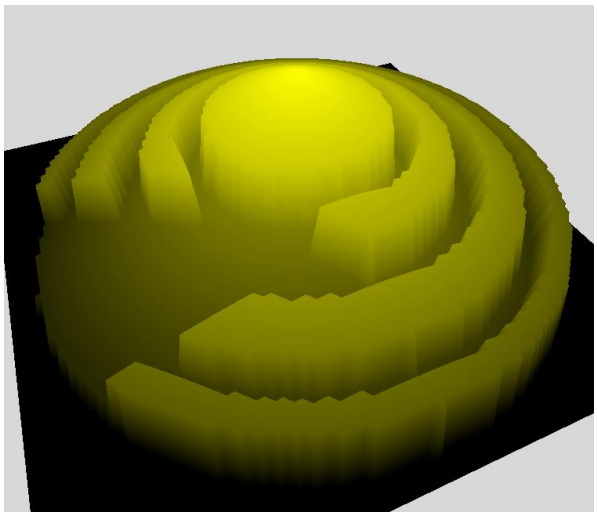
Lara “enterrada” en mapa 2:

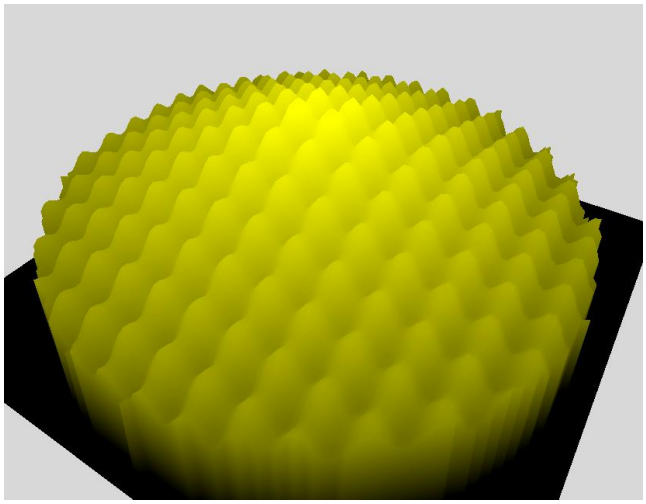


Lara “flotando” en mapa 2:

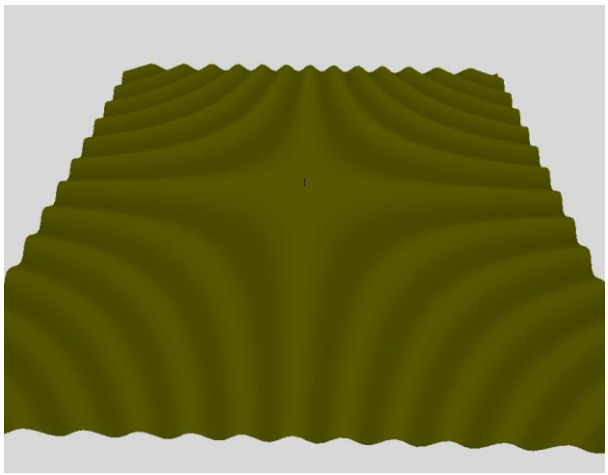


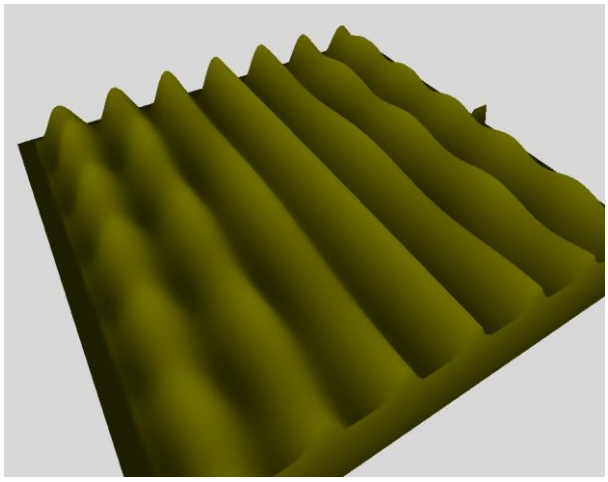
Suelo y cielo de mapa 2:





Suelo y cielo de mapa 1:





Autoevaluación

Criterio-Puntaje	0	1	2	3
OpenGL		x		
Shaders		x		
Modelos geométricos		x		
Transformaciones			x	
Texturas			x	
Modelación jerárquica		X		

Curvas	x			
Vistas y proyecciones			x	
Iluminación local				x
Mallas geométricas				X
Lógica de la aplicación			X	
Control de usuario			x	
Visualización de estado			x	----------