Departamento de Ciencias de la Computación
Facultad de Ciencias Físicas y Matemáticas
Universidad de Chile
CC3501-1 Modelación y Computación Gráfica

TAREA N° 2

Juego interactivo "Lara Jones"

Integrantes: Nahuel Gómez Raguileo
Profesor: Daniel Calderón
Auxiliares: Nelson Marambio
Sebastián Olmos
Ayudantes: Alonso Utreras
Beatriz Graboloza
Heinich Porro Sufan
Nadia Decar
Tomás Calderón
Fecha de entrega: 13 de junio de 2021
Santiago, Chile

Resumen

En el presente documento se abordará la solución implementada para la tarea 2.a *"Lara Jones"* mediante el uso del lenguaje de programación *Python*, la interfaz *OpenGL*, y los diversos conocimientos adquiridos en cátedras y clases auxiliares, de estos, las más útiles, son la utilización de mallas poligonales para modelar el terreno que se generará para el recorrido del personaje, y la iluminación implementada para la representación de la linterna. Agregado a lo anterior, se continuó trabajando con "clases" y una metodología dividida en tres partes, "La lógica", "La gráfica" y "El controlador".

Para no caer en la monotonía del problema de Lara, se utilizó el sprit de personaje genérico, el cual deberá conseguir todas las piezas de oro para poder descifrar el acertijo y así escapar de la cueva en la que cayó por accidente. ¡Por favor no te salgas de los parámetros de la cueva, podrías caer en un gran Lío!

Índice de Contenidos

1. Solución Propuesta **1**
1.1. Mallas 1
1.2. Iluminación 1
1.3. Otros 1

2. Instrucciones de Ejecución **2**

3. Resultados **3**

4. Autoevaluación **5**

Lista de Figuras

1. Foto de nodos de la pieza 1
2. Foto de la pieza que se debe conseguir en el juego 3
3. Foto general de la cueva con luminosidad ambiental baja 3
4. Foto de la pieza a distancia 4
5. Foto de victoria 4

1. Solución Propuesta

Las dos principales estrategias usadas para realizar la solución del problema fueron la utilización de “Mallas” e “iluminación local´´:

1.1. Mallas

Se decidió trabajar con “OpenMesh” (www.graphics.rwth-aachen.de/media/openmesh_static/Documentations/OpenMesh-6.2-Documentation/index.html) debido a las facilidades que entrega para maniobrar con los vértices y caras de un grafo. A través de de la libreria se logró implementar de manera eficiente texturas para la cueva de nuestro juego. La función “*_create*, la cual se encarga de modelar el piso y el techo, según el parámetro que le indiquemos (el piso viene dado por “0”, mientras que el techo por “1”).

Además tenemos la función *toShape* la cual es encargada de tranformar las mallas en una “Shape” (modificable mediante matrices), a la cual podemos agregarle texturas.

1.2. Iluminación

Se pidió en el enunciado de la tarea proporcionar al personaje una linterna de distintas intensidades, la cual permitiera ubicarse de mejor manera en la oscuridad de la cueva, es por esto que se decidió trabajar con iluminación local. Para implementarla, se aprovecharon las facilidades que entrega “OpenMesh” para obtener las normales de vértices y caras de polígonos. Sin embargo, no se logró el resultado esperado; la idea general fue modificar el archivo “*lighting_shaders.py*”, específicamente la versión “Phong” del mismo.

1.3. Otros

Además; se trabajó con modelación jerárquica y curvas para la realización de la pieza del rompecabezas:

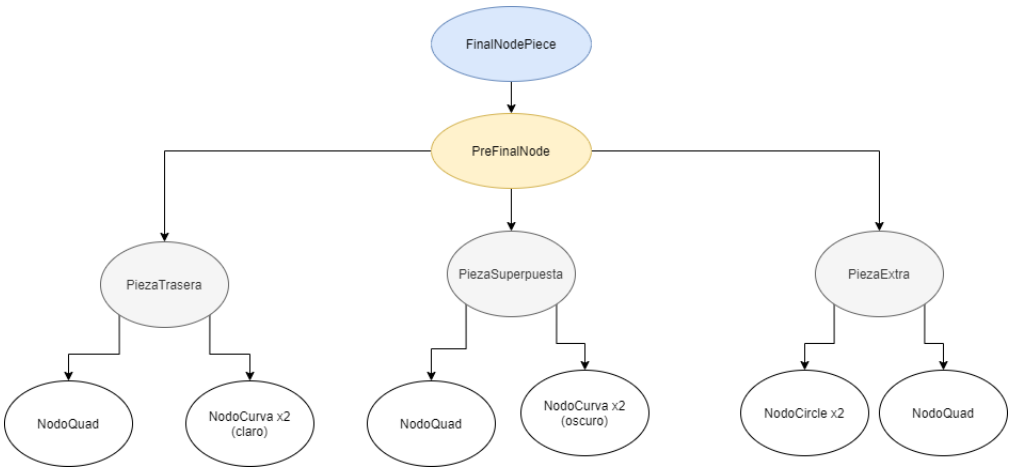

Figura 1: Foto de nodos de la pieza

También se utilizó una metodología de trabajo para volver más amena la interfaz de ejecución, la cual está conformada por tres grandes apartados, la vista del juego (encargada del apartado gráfico), el modelo del juego (guarda las clases y modelos de personajes y objetos), y el controlador (encargado de las condiciones lógicas del juego); se ocupó un sprite genérico (https://www.pngegg.com/en/png-zwnbo) con distintas posiciones de movimiento, otorgándole mayor credibilidad al caminar de este.

2. Instrucciones de Ejecución

El programa será llamado con la entrada:

python cave.py map.npy textures.png N

- **cave.py :** Ejecuta el código.
- **map.npy :** Corresponde al mapa de la cueva, será modelado mediante la utilización de mallas poligonales. Se entregaran dos cuevas prefabricadas ("cueva1.npy" y "cueva2.npy").
- **textures.png :** Son las texturas que se le aplicarán a la cueva. Debido a dificultades de implementación con las mallas, "textures.png" corresponderá a una textura de un único tipo, la cual modelará toda la cueva (se entregó la textura descrita en el enunciado dividida en 12 partes para facilitar el testeo).
- **N :** Número de piezas doradas que irán apareciendo, cuando se consigan todas se ganará el juego.

  Para el movimiento del personaje se utilizará el mouse para girar la vision nuestra y del personaje, además, con click derecho e izquierdo se podrá avanzar y retroceder.

3. Resultados

A continuación se mostrarán los principales resultados de la implementación del juego.

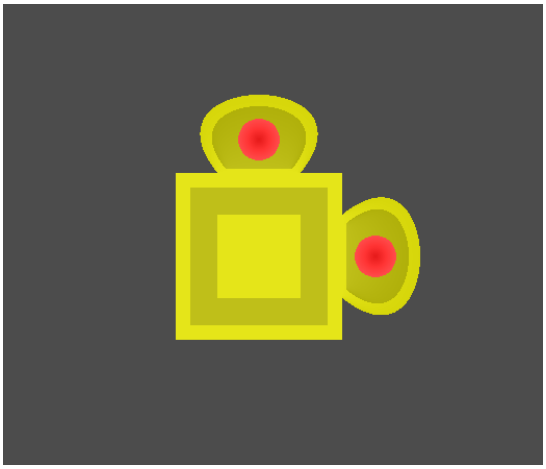

Figura 2: Foto de la pieza que se debe conseguir en el juego

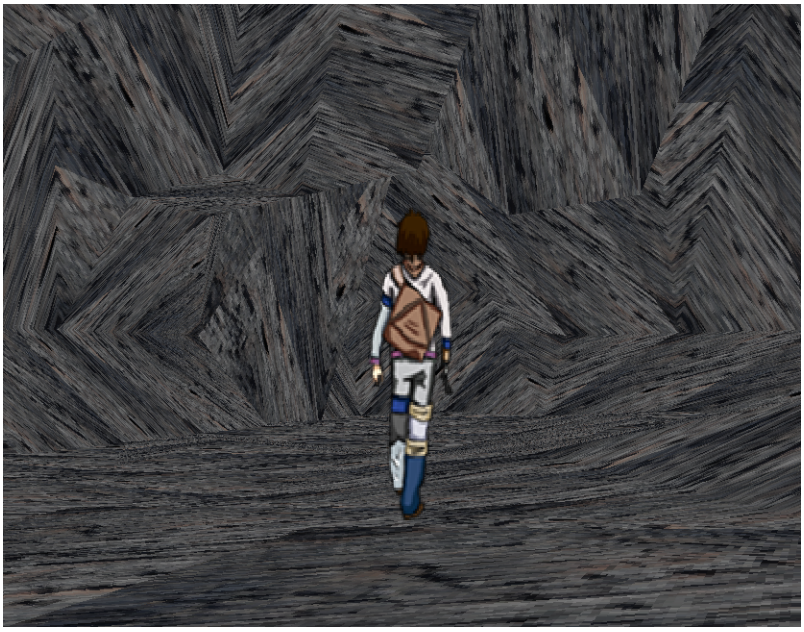

Figura 3: Foto general de la cueva con luminosidad ambiental baja

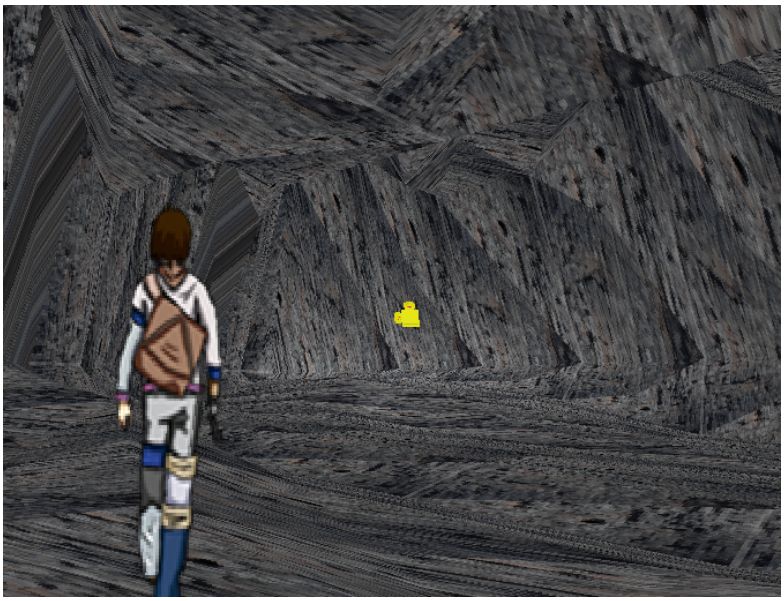

Figura 4: Foto de la pieza a distancia

Figura 5: Foto de victoria

4. 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 poligonales			X	
Funcionalidades mecánicas		X		
Entrada o control de usuario		X		
Visualización del estado del programa		X		

Nota: Hubo muchas implementaciones que estuve mucho tiempo intentado implementar y que sin embargo no se vieron reflejadas, estas son principalmente las correspondientes a iluminación. A modo personal, siento que la tarea fue bastante más profunda en comparación a los conocimientos adquiridos en cátedras y auxiliares.