Tarea 3 – Shaders

Profesora: Nancy Hitschfeld K.
Auxiliar: Vicente González
Ayudante: Juan Flores

Descripción de la tarea

Esta tarea se centra en comprender el funcionamiento general de los shaders en el contexto de la visualización gráfica por computadora. El objetivo es implementar un visualizador de shaders en cualquier plataforma (Javascript/Three.js, C++/GLFW, Java/LWJGL, Python/GLFW, C/GLUT, etc.) donde se resuelva un problema específico, como la implementación de la superficie de un mar con oleaje, la visualización de un terreno, la animación de un fractal, la visualización de objetos con diferentes shaders, generación de ruido celular u otra aplicación interesante que encuentre.

Además de la implementación, se solicita enviar un breve video de 3-4 minutos de su proyecto en funcionamiento y una presentación en formato PDF con un máximo de 10 diapositivas en las que explique la implementación del trabajo y cómo resolvió el problema (técnicas, metodología, algoritmos, etc.).

A continuación se presentan las opciones entre las cuales debe elegir una. Estos son los requisitos mínimos solicitados, pero se premiar á la innovación. Debe entregar todos los archivos necesarios e instrucciones de cómo ejecutarlo a través de U-Cursos.

1 Cellular Noise

En esta opción de la tarea se solicita implementar el Cellular Noise, que es un tipo de ruido que se puede observar en la Figura 1. Se requiere desarrollar una aplicación que ofrezca una interfaz para ajustar y jugar con todos los parámetros necesarios.

El Cellular Noise es un tipo de ruido utilizado en muchos estudios, junto con otros como Perlin Noise y Turbulence Noise. Para comprender mejor este tipo de ruido y su implementación, se puede consultar el siguiente enlace: https://thebookofshaders.com/12/

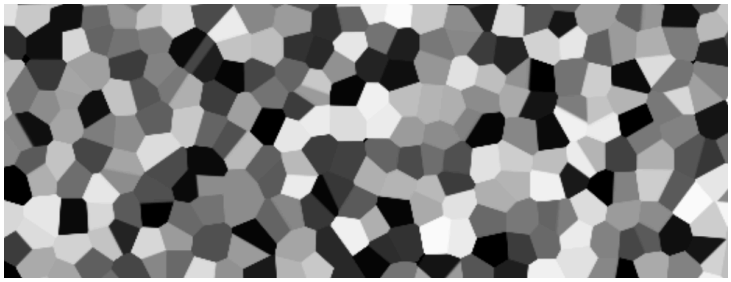

Figura 1: Ejemplo de ruido celular

La implementación debe permitir al usuario explorar diferentes configuraciones y ajustes de los parámetros del Cellular Noise. Se recomienda proporcionar controles intuitivos en la interfaz para cambiar parámetros como la escala, el contraste, la frecuencia, el tipo de celda y cualquier otro parámetro relevante para obtener diferentes efectos visuales a partir del ruido celular.

2 Curvas de nivel de un terreno

El dibujo/rendering de superficies que representan un terreno es muy útil en juegos y aplicaciones científicas que requieren cálculos sobre estas superficies. El terreno est á definido por puntos y triángulos que los conectan. La coordenada Z corresponde a la altura del terreno, que puede variar entre 0 y 6000 metros. Para esta tare a se pide representar un terreno como curvas de nivel. Como se ve en la Figura 2.

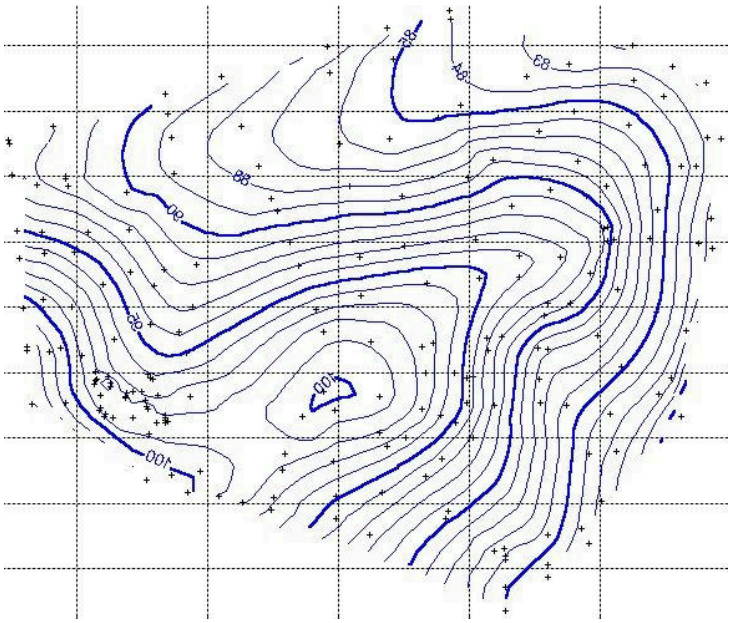

Figura 2: Curvas de nivel

En esta opción, se solicita crear un terreno utilizando una función fractal o cargarlo desde un archivo con al menos 10000 puntos. Además, se debe permitir realizar zoom y un-zoom en el terreno, crear una cámara que permita visualizar el terreno desde todas las perspectivas y definir una fuente de luz puntual y ambiental para lograr un aspecto más realista. También se requiere implementar un vertex shader, un fragment shader y un geometry shader.

El vertex shader debe permitir cambiar la posición de los puntos del terreno para genera r una nueva superficie cada vez que el usuario presione una tecla elegida por ustedes. Pueden seleccionar cualquier función que consideren adecuada para este propósito.

El fragment shader debe pintar cada curva, de tal manera, que las curvas de cada nivel sean distintas.

El usuario debe tener la opción de definir la altura en la dibujara cada curva de nivel, el geometry shader debe hacer que todos los triángulos que intersecten dicha altura debe generar un segmento.

Se debe proporcionar una interfaz que permita al usuario interactuar con el terreno, como realizar zoom, cambiar la perspectiva de la cámara y explorar diferentes configuraciones de iluminación.

3 Proyecto Personal

Esta opción está destinada a aquellos que no estén satisfechos con ninguna de las opciones anteriores. Para solicitar esta opción, deben enviar un correo al profesor auxiliar explicando su

proyecto, alcance y objetivos. Se espera que la dificultad de su proyecto esté a la altura de las propuestas presentadas en este documento. El correo debe incluir la siguiente información:

- Título de la propuesta.
- Descripción del shader o aplicación a desarrollar.
- Referencia a utilizar.

La fecha límite para enviar la propuesta es el **16 de Julio a las 23:59**. En esta opción, tienes la libertad de proponer y desarrollar tu propio proyecto relacionado con shaders, utilizando el vertex shader, fragment shader y geometry shader, y aplicaciones gráficas. Puedes enfocarte en áreas específicas que te resulten interesantes, como técnicas avanzadas de renderizado, simulaciones interactivas, visualizaciones científicas, entre otros.

Información importante

- Se debe incluir un archivo README que explique cómo ejecutar su programa. Si no se incluye un README, su tarea ser á evaluada con la nota mínima.
- Recuerde incluir un vídeo de 3-4 minutos que muestre su aplicación/shader en funcionamiento. El vídeo puede estar en cualquier formato reproducible y debe adjuntarse con la entrega. Si no incluye un vídeo o demo, su tarea ser á evaluada con la nota mínima.
- Recuerde incluir una presentación de 5-10 láminas que explique lo más importante de su shader e implementación, los problemas encontrados y las soluciones. Piense en la mejor manera de enseñar el shader a alguien externo que quiera implementarlo. Si no incluye la presentación, su tarea ser á evaluada con la nota mínima.