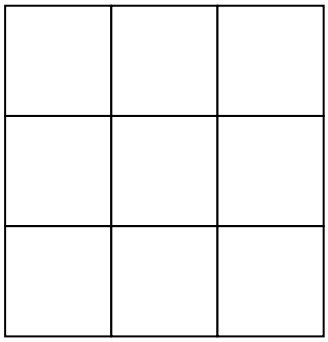
Universidade Federal de Santa Maria
Curso de Ciência da Computação
Disciplina: Computação Gráfica Avançada
Segundo Semestre de 2023
Prof. Cesar Tadeu Pozzer
Data: 18/09/2023

Trabalho 1 – Shader

Objetivos: explorar Vertex shader, tesselation shader, Gemetry shader, fragmente shader e iluminação.

Desenvolva um programa em C++, utilizando a API OpenGL 4.x, para simular um terreno formado por 9 patches (**ou mais**), dispostos como na seguinte figura.



1. Cada patch deve ser refinado com tesselation shader usando um diferente nível de refinamento.
2. Os vértices nas bordas de patches vizinhos devem ser iguais (não pode formar t-vertex).
3. Deve-se usar o geometry shader para gerar as normais de cada triângulo gerado.
4. Deve-se utilizar uma textura (imagem) ou função procedural (ex: função noise) para fazer a perturbação de altura do terreno em cada vértice (displacement mapping).
5. O terreno deve ser iluminado (fragment shader) por duas fontes luminosas em movimento.
6. O nível de refinamento deve mudar em função da distância da câmera.
7. Deve-se poder visualizar o terreno em wireframe e com preenchimento.
8. Deve-se também poder ajustar o nível de refinamento do terreno (de um patch em específico – pode ser o central, e os vizinhos devem se ajustar para evitar a formação de T-vertex).

Explorem formas de passar a resolução dos patches para o shader.

Extras:

1. Explorem a geração de t-vertex para ver o efeito causado.
2. Usar o nível de variação do heightmap para definir a taxa de refinamento.
3. Animação do terreno, como se fosse um mar.
4. Usem a criatividade.

Data e Formato de Entrega

- O trabalho deve ser apresentado durante a aula.
- O trabalho deve ser enviado pelo classroom.
- Deve-se enviar fontes e o projeto para o compilador **Visual Studio 2022 Comunity**. Envie **todos os arquivos necessários para a compilação. Não** enviar pasta .vs (pasta oculta), nem binários, arquivos obj, etc.
- O programa deve ser enviado em um arquivo compactado fulano.rar (fulano = login ou nome do aluno). Dentro deste arquivo deve haver um diretório com o mesmo nome do arquivo e dentro deste diretório os arquivos do trabalho.
  Observe o tamanho do arquivo .rar enviado. Não pode ser maior que 15 MB. Observe o tamanho da pasta antes de compactar.
- Ex: o arquivo pozzer.rar deve conter um diretório chamado pozzer, e dentro do diretório devem estar os arquivos do trabalho.

Critério de Avaliação

- Documentação: descrever no cabeçalho de cada arquivo a ideia geral do código e detalhes específicos de partes que mereçam uma explicação – não comente por exemplo o que faz b++.
- README.txt: incluir um arquivo "README.txt" contendo informações sobre quais funcionalidades foram implementadas (requisitos e extras).
- Pontualidade: Trabalhos não entregues na data não serão avaliados e receberão nota zero.
- Legibilidade: nome de variáveis, estruturação do código. O código digital a ser entregue deve ter 4 espaços de identação e não deve possuir tabulações.
- Clareza: facilidade de compreensão – evite códigos complexos e desnecessários. Adote a solução mais simples possível.
- Funcionalidade: o programa deve satisfazer todos os requisitos. Programas que não compilarem ou que não atenderem nenhum requisito receberão nota 0 (zero).

Você pode discutir estratégias e ajudar o colega na implementação, porém evite passar código fonte. Programas semelhantes terão nota 0 (zero).