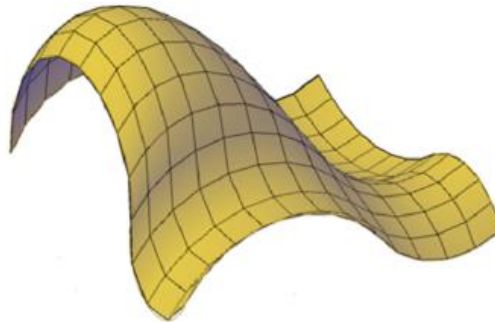


Universidade Federal de Santa Maria
Curso de Ciência da Computação
Disciplina: Computação Gráfica Avançada
Segundo Semestre de 2015
Prof. Cesar Tadeu Pozzer
Data: 27/10/2015

Trabalho 3 – Iluminação com Shader



Descrição:

Baseado no demo `gsl_2_GLSL33_VAO`, desenvolva um programa em C++, utilizando a API OpenGL, para renderizar uma superfície não plana com pelo menos 10x10 vértices. O programa deve fazer a iluminação difusa e especular. O terreno deve ser modelado usando VAO e VBO. Deve ter pelo menos uma fonte de luz móvel no cenário. Deve-se utilizar as transformações da classe `myGL`.

Deve-se observar os seguintes requisitos:

- Deve ter controle de FPS – Deve rodar na mesma velocidade em qualquer máquina;
- Utilizar conceitos de vetores e suas operações – produto escalar, etc;
- Deve permitir mover a câmera no cenário.
- A iluminação deve ser feita de forma correta.
- Deve-se fornecer um relatório simples descrevendo as técnicas implementadas. Esse relatório pode estar dentro do arquivo `main.cpp`.

Data e Formato de Entrega:

- Data: 06/11/2015.
- No email e no cabeçalho do arquivo, devem conter o nome completo e matrícula do aluno. O arquivo deve ser enviado para `pozzer3@gmail.com`, `afrasson@inf.ufsm.br`, `tengel@inf.ufsm.br` com o subject “CGA T3”.
- 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.

- **Ex:** o arquivo **pozzer.rar** deve conter um diretório chamado **pozzer**, e dentro do diretório devem estar os arquivos do trabalho, como por ex: **math.dev**, **math.cpp**, **math.h**, **aux.cpp**, ...

Critério de Avaliação:

- **documentação:** descrever no cabeçalho de cada arquivo a idéia geral do código e detalhes específicos de partes que mereçam uma explicação – não comente por exemplo o que faz **b++**.
- **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 **3 espaços de indentação** 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 a nota 0 (zero).