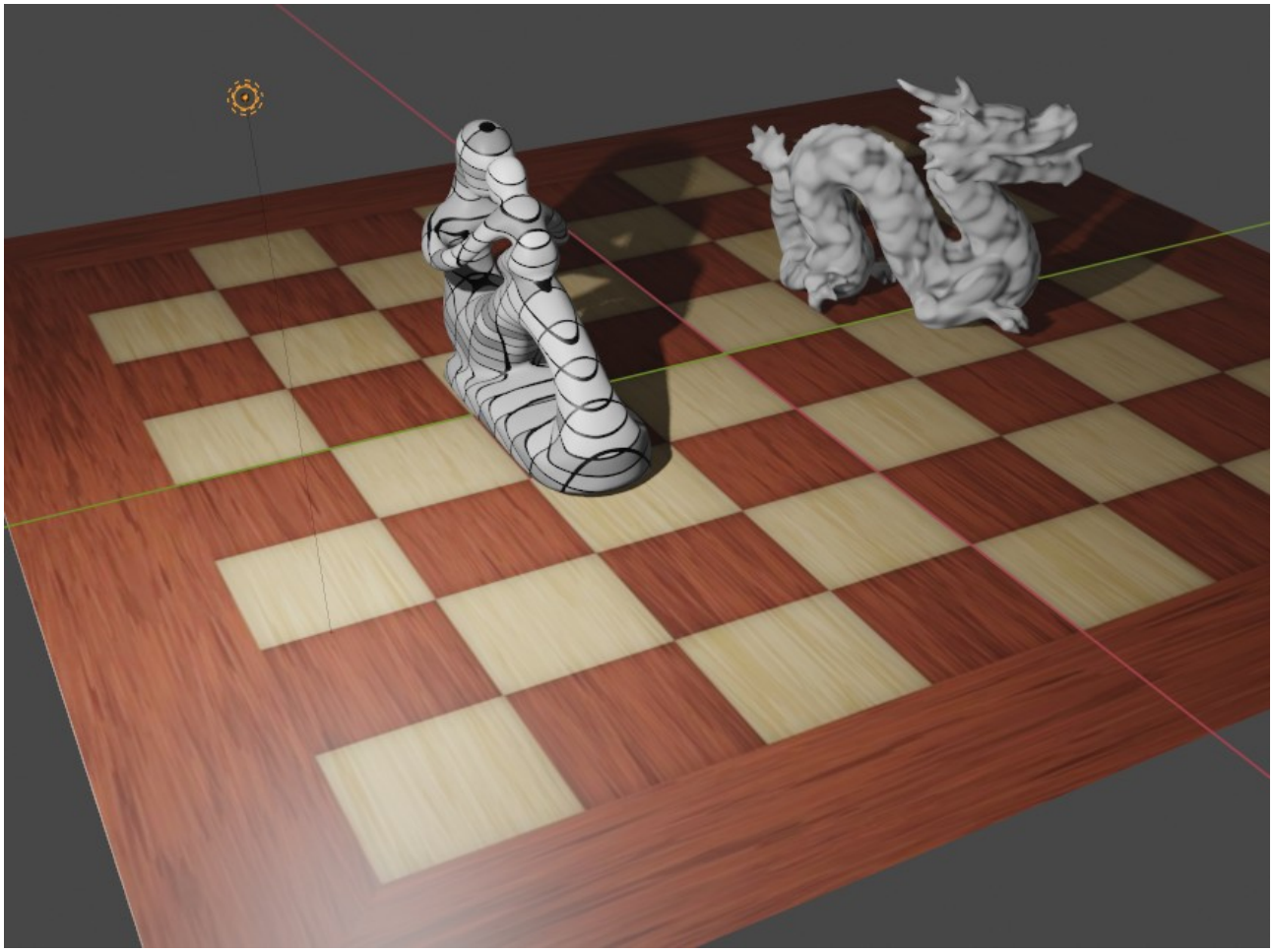
Universidade Federal do Ceará
Computação Gráfica
Prof. Rafael Ivo

Trabalho Individual 2 – Cenário 3D

Descrição

Implemente um programa de montagem de cenários 3D, com objetos montados sobre um cenário.



Obs: A iluminação aplicada na figura acima possui alguns efeitos que não precisam ser implementados porque são bastante complexos, como sombra projetada.

O programa deve conter as seguintes funcionalidades:

1. Adicionar Objetos

- Os objetos devem ser malhas triangulares lidas de arquivos OBJ.
- Cada objeto deve ser movido, rotacionado e/ou escalonado para ajustá-lo ao cenário. As transformações podem estar dentro do código, mas sendo utilizadas corretamente.
- Cada objeto deve ter propriedades de material para a iluminação (ambiente, difusa e especular).
- A cena deve ser composta de, no mínimo, dois objetos e um cenário (pelo menos um chão). Para o melhor resultado da iluminação desse plano, faça um tabuleiro como no código disponibilizado à turma. Mas você pode criar cenários mais elaborados e com mais objetos.

2. Iluminação

- O modelo de iluminação a ser implementado é o Modelo de Iluminação de Phong.
- A fonte de luz básica é a luz pontual. Mostre algum elemento a fonte de luz está para ficar facilitar o usuário posicioná-la (similar à imagem acima).
- A fonte de luz também deve possuir propriedades de iluminação de Phong (ambiente, difusa e escpecular)
- O sombreamento deve ser Gouraud.

3. Textura

- Os objetos devem possuir textura.
- Você é livre para definir o mapeamento de textura. Mas deve explicar qual foi o método usado.
- Alguns modelos OBJ trazem as coordenadas de textura e você pode utilizá-las. Entretanto, em pelo menos um único objeto, você deve calcular as coordenadas de textura. No exemplo acima, o mapeamento realizado na escultura foi o mapeamento planar.

4. Outras Informações

- Você é livre para definir a câmera da forma como desejar, mas o programa deve permitir visualizar o cenário de vários ângulos diferentes.
- Toda a interface para realizar as funcionalidades acima não será cobrada. Os dados dos objetos e da fonte de luz podem ser adicionados por:
  - interface gráfica (botões, menus, etc.)
  - mouse e/ou teclado
  - arquivo TXT
  - dentro do código
- Atenção! A interface não é importante, mas as funcionalidades devem ser generalizadas. Ou seja, não devem funcionar apenas em casos específicos. Então se você definir as transformações geométricas dos objetos, por exemplo, dentro do código, faça de forma que seria fácil expandir o código para que essas transformações pudessem ser feitas por alguma interface mais avançada.

Regras Gerais

- O envio do trabalho será exclusivamente no SIGAA, no prazo estipulado.
- Utilize boas regras de programação, como identação de código, variáveis intuitivas, modularização (funções e classes) e vários comentários, explicando as ideias utilizadas.
- Cada arquivo do código fonte deve possuir no início o seu nome e número de matrícula.
- Crie um vídeo explicando seu trabalho. A explicação é de extrema importância. Sua ausência pode implicar na exclusão do trabalho. Não é necessário enviar o arquivo de vídeo, mas um link para visualização na descrição do trabalho ao enviar. Coloque o link no comentário da atividade do SIGAA.

Apêndice - Arquivo OBJ

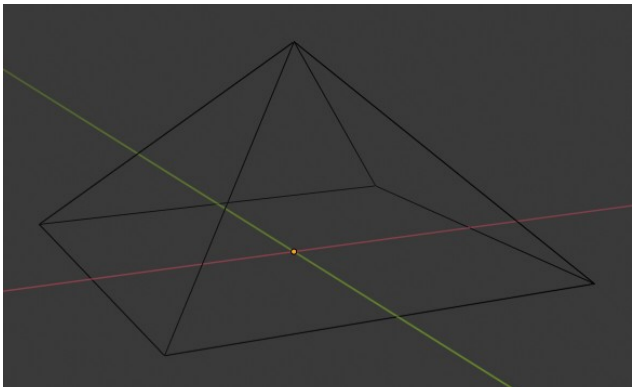
Um arquivo OBJ é um arquivo de texto contendo as informações que descrevem uma malha poligonal. Cada arquivo OBJ possui uma estrutura similar a seguinte:

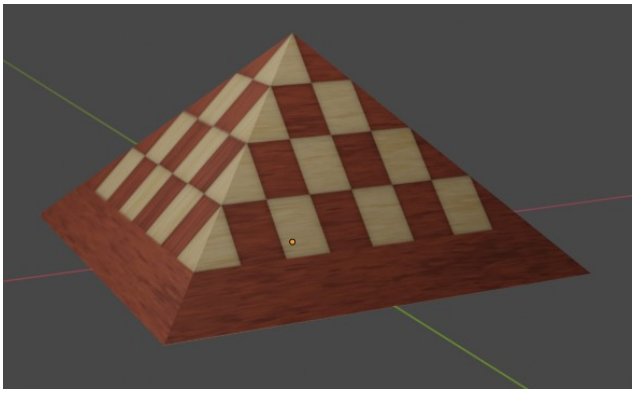
```
# lista de vértices
v x y z w
.
.
.
# lista de coordenadas de textura (inica
vt s t u v
.
.
.
# lista de vetores normais
vn nx ny nz nw
.
.
.
# lista coordenadas paramétricas
vp u v w
.
.
.
# lista de faces (a quantidade de vértices associados a uma face é variável)
f va/vta/vna vb/vtb/vnb vc/vtc/vnc …
.
.
# lista de vértices de uma polilinha
l 5 8 1 2 4 9
```

Atenção!

- Qualquer coisa escrita nesse arquivo após o caractere ‘#’ é descartado, assim como em comentários num programa.
- A única seção obrigatória em um arquivo OBJ é a lista de vértices. Todas as demais são opcionais. Entretanto, na maioria dos modelos estão presentes a lista de vértices e a lista de faces, e em alguns há a lista de vetores normais. Todas as listas são numeradas com índices iniciando em 1.
- Os trechos em vermelho são opcionais. Alguns arquivos possuem, outros não.
- Na lista de faces, cada face é definida com os índices dos vértices (opcionalmente com índices dos vetores normais e das coordenadas de textura separados por barra)
- Para mais detalhes, procure na internet sobre o formato de arquivo OBJ. É um formato muito conhecido e com muito material.

Abaixo está um exemplo de um arquivo OBJ de uma pirâmide.





```
#vértices
v -2.0 0.0 -2.0 #vertices da base
v 2.0 0.0 -2.0
v 2.0 0.0 2.0
v -2.0 0.0 2.0
v 0.0 2.0 0.0   #vertices do topo

#coordenadas de textura de cada vértice na mesma ordem
vt 0.0 0.0
vt 1.0 0.0
vt 1.0 1.0
vt 0.0 1.0
vt 0.5 0.5

#faces (como vetores normais não estão presentes, eles não aparecem)
f 1/1/ 5/5/ 2/2/
f 2/2/ 5/5/ 3/3/
f 1/1/ 2/2/ 3/3/ 4/4/
f 3/3/ 5/5/ 4/4/
f 4/4/ 5/5/ 1/1/
```

Obs: Arquivos OBJ permitem faces com quaisquer quantidade de vértices (na pirâmide acima, a base é um quadrilátero). Entretanto, para este trabalho, assuma que o modelo possui apenas faces triangulares. Se você baixar um modelo com faces não triangulares você pode adotar as soluções:

1. Usar um modelador 3D para dividir as faces não triangulares em triângulos;
2. Implementar a adição de mais de uma face ao mesmo tempo caso leia mais de três vértices em uma face, dividindo automaticamente
3. Descartar do quarto vértice em diante. (Fará o modelo ter buracos ao exibir. Evite.)