UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
Computação Gráfica Avançada – 2º semestre de 2015.
Renato Ferreira e Erickson Nascimento

Trabalho Prático – Ray Tracing

O objetivo desse trabalho é exercitar conceitos de iluminação global através da produção um programa que realiza a renderização por ray tracing de uma cena descrita em um arquivo cujo formato está descrito a seguir.

Arquivo de Descrição de Cena

A descrição da cena é feita por um arquivo texto com 5 sessões bem definidas, a dizer:

1) Descrição de Câmera

A descrição da câmera que captura a cena descrita no arquivo está nas 4 primeiras linhas do arquivo. As 3 primeiras linhas contém 3 grandezas de ponto flutuantes cada uma, que correspondem, respectivamente, às coordenadas da posição central da câmera (coordenadas do olho), às coordenadas da posição para onde a câmera está voltada (coordenadas do centro do plano de projeção) e finalmente as coordenadas de um terceiro ponto, na direção do qual a normal da câmera está apontando, descrevendo portanto a orientação da câmera. Na quarta linha do arquivo, temos uma grandeza de ponto flutuante que representa a abertura da câmera em relação ao eixo Y. Esse quarto parâmetro descreve sumariamente a janela visível no plano de projeção.

2) Descrição de Luzes

A próxima linha do arquivo de configuração, depois da descrição da câmera deverá conter um número inteiro, maior do que zero, representando quantas luzes estão presentes na cena. Em seguida, cada uma dessas luzes é detalhada, sendo uma em cada linha subsequente. Para cada luz, a descrição consistirá de 9 grandezas de ponto flutuante a dizer: as 3 coordenadas da fonte de luz, a cor da luz emitida pela fonte, em formato RGB (com magnitude entre 0 e 1) e 3 parâmetros de atenuação da respectiva fonte de luz, sendo o primeiro parâmetro um coeficiente constante de atenuação, o segundo a atenuação proporcional à distância da fonte da luz e finalmente o coeficiente de atenuação proporcional ao quadrado da distância da fonte da luz. A primeira fonte de luz descrita no arquivo é a luz ambiente, e suas coordenadas estão presentes na descrição meramente para simplificar o formato do arquivo.

3) Descrição de Pigmentos

Após a definição de cada uma das fontes de luz, o arquivo texto conterá uma nova linha contendo um número inteiro positivo com um número de pigmentos presentes na cena. Esses pigmentos são por sua vez descritos, um em cada linha subsequentes. O programa deve comportar pelo menos 3 tipos distintos de pigmentos, que são determinados por uma string no início de cada linha, a dizer: *solid*, *checker* e *texmap*. O pigmento *solid* consiste em uma cor sólida, que é especificada pelos valores RGB (entre 0 e 1). O pigmento *checker* corresponde a uma textura procedural que produz um padrão quadriculado em 3d. Para esse padrão são especificados duas cores (também em RGB, entre 0 e 1) e uma sétima grandeza de ponto flutuante que corresponde ao comprimento dos lados do cubo do padrão. Finalmente, o pigmento *textmap* corresponde a

uma textura básica. Como argumentos para esse pigmento, além do arquivo contendo a textura propriamente dita, em formato ppm, são passados 8 grandezas de ponto flutuante. Para se transformar as coordenadas de um ponto 3D para as coordenadas 2D da textura usa-se combinação linear, ou seja 4 números de ponto flutuante são usados para se calcular cada coordenada da textura. Em outas palavras, são dois vetores de 4 elementos, P0 e P1 passados como argumento para a textura e dado um ponto 3D em coordenadas homogêneas, PC, a cor do ponto PC será dada pela cor da coordenada (s, r) na textura onde s = P0•PC e r = P1•PC.

4) Descrição de Acabamentos

Em seguida, o arquivo texto trás a descrição dos acabamentos de superfícies existentes na cena, seguindo o mesmo formato: uma linha contendo um inteiro com o número de acabamentos, e cada acabamento numa linha subsequente. A descrição de cada acabamento é bastante simples e consiste de 7 valores de ponto flutuante a dizer: *ka* (coeficiente de luz ambiente), *kd* (coeficiente de luz difusa), *ks* (coeficiente de luz especular), α (expoente para reflexão especular). Esses 4 primeiros parâmetros tem a ver com as propriedades de interação da superfície com as fontes de luz, e são parecidas com as propriedades do modelo de Phong. Os próximos 3 parâmetros já são relacionados com o algoritmo de ray tracing propriamente ditos a dizer: *kr*, *kt* e *ior*. O primeiro é o coeficiente de reflexão (*kr*) e se for maior do que zero, significa que a superfície reflete luz e o raio refletido deve ser seguido e a cor obtida somada à cor final, ponderada pelo coeficiente. Já o *kt* é o coeficiente de transmissão, e em valor maior do que zero, significa que a superfície transmite luz e o raio transmitido deve ser seguido e a cor obtida somada à cor final ponderada por esse coeficiente. Para o cálculo do raio transmitido usa-se a lei de Snell onde é necessário o terceiro parâmetro, que é a taxa entre os índices de refração do ambiente (*n1*) e o índice de refração do material (*n2*), ou seja, *ior = n1/n2*.

5) Descrição de Objetos

Finalmente, o arquivo trás a descrição das superfícies que constituem a cena. Novamente o formato é o mesmo: um inteiro contendo o número de superfícies presentes na cena, descritas uma em cada linha subsequente. A descrição de cada superfície consiste de um inteiro com referência ao pigmento da superfície, seguido de outro inteiro com referência para o acabamento dela. Em seguida, o tipo de superfície é descrita, podendo ser de dois tipos: esferas e poliedros convexos. As esferas são descritas pelo string “sphere” seguido de 4 números de ponto flutuante, a dizer, as coordenadas (x, y, z) do centro da esfera, e o raio. Já os poliedros convexos são descritos pela string “polyhedron”, seguido de um inteiro para o número de faces do poliedro e, a seguir, cada face em uma nova linha, os coeficientes da equação do plano que contém a face. O poliedro convexo seria definido pela interseção dos semi-espaços definidos pelos planos das faces.

O Que Deve Ser Entregue

O seu trabalho é escrever um programa que receba como parâmetros de linha de comando obrigatórios contendo o nome de dois arquivos. O primeiro o arquivo de entrada, conforme descrito acima e ilustrado na Figura 1. O segundo é um arquivo em formato ppm que deve ser produzido a partir da leitura do arquivo de entrada, conforme mostrado na Figura 2. Dois parâmetros de linha de comando opcionais devem ser aceitos pelo programa, com a especificação das dimensões largura e altura da imagem a ser produzida. Caso esses parâmetros não sejam passados, a imagem gerada deve ter dimensões (800x600).

```
0    30   -200
0    10   -100
0     1     0
40
3
0     0     0   1  1  1    1  0  0
60.0 160.0 -200.0   1  1  1    1  0  0
-80.0 160.0 -200.0   1  1  1    1  0  0
3
texmap       rainbow1.ppm
0 .001   0   .12
0    0   0   0
checker      .08  .25  .20     .93  .83  .82    40
solid        1    1    1
3
0.80 0.00 0.00     1   0.0  0   0
0.30 0.40 0.00     1   0.3  0   0
0.11 0.11 0.30   1000   0.7  0   0
```

```
12
0 0 sphere        0      0      0    600
2 2 sphere        0   32.7      0     20
2 2 sphere     -5.98      0 -22.31     20
2 2 sphere    -16.32      0  16.32     20
2 2 sphere     22.31      0   5.98     20
2 2 sphere      5.98 -32.66  22.31     20
2 2 sphere     16.32 -32.66 -16.32     20
2 2 sphere    -22.31 -32.66  -5.98     20
2 2 sphere    -11.95 -32.66 -44.61     20
2 2 sphere    -32.66 -32.66  32.66     20
2 2 sphere     44.61 -32.66  11.95     20
1 1 polyhedron 5
0  1  0    60
1  0  0  -300
-1  0  0  -300
0  0 -1  -300
0  0  1  -300
```

Figura 1: Arquivo de entrada de exemplo com a descrição de uma cena pra ser renderizada.

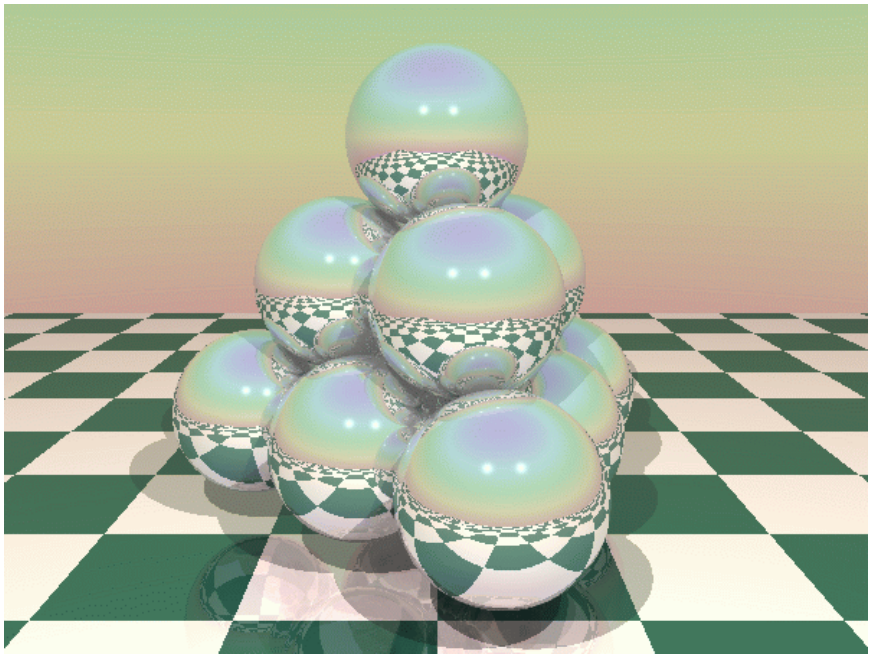

Figura 2: Imagem produzida pelo programa para o arquivo de entrada de exemplo.

Avaliação

A funcionalidade especificada acima vale 70% da nota final do trabalho, sendo considerada funcionalidade básica. Para se obter o restante dos pontos do trabalho (ou até mesmo mais pontos extras) funcionalidades adicionais podem/devem ser implementadas no renderizador. Essas funcionalidades serão avaliadas conforme a dificuldade da implementação, o efeito obtido com ela, e a qualidade da implementação. Exemplos de funcionalidades extras com suas respectivas pontuações:

1) Adição de novas superfícies para renderização, por exemplo outras cônicas, além da esfera (5%) ou sólidos em formato CSG (10%).
2) Implementação de ray tracing distribuído para geração de sombras suaves, acrescentando uma área nas fontes de luz (5-10%), ou reflexões ou refrações imperfeitas, acrescentando parametros de imperfectibilidade dos acabamentos (5-10%).
3) Utilização ray tracing distribuído para implementação de modelos de câmera com abertura e distância focal variadas, acrescentando-se parametros de lente na descrição da câmera (10%).
4) Utilização de ray tracing distribuído para incorporação de aspectos temporais e produção de imagens com corpos em movimento, acrescentando vetores velocidade nos objetos e um tempo de exposição na câmera (15%).
5) Outras idéias, como anti-aliasing adaptivo, ou outras estratégias de espalhamento dos raios (15%).

O trabalho deve ser individual. Podem discutir idéias e etc mas cada um deve ter a sua implementação independente dos colegas. Deve ser entregue um arquivo .tar.gz ou .zip contendo todo o programa fonte, com os Makefiles e bibliotecas necessárias para a compilação do programa. Nesse pacote deve haver também um arquivo README com uma descrição da sua implementação, e as instruções para a compilação e a execução do seu jogo.

Qualquer dúvida, entre em contato comigo. Ou acrescente a sua interpretação no arquivo README, e mãos a obra.

Bom trabalho,
Renato e Erickson.