TP2: Raytracer
Computação Gráfica

Pontifícia Universidade Católica de Minas Gerais

Prof.: Alessandro Ribeiro da Silva

20 de maio de 2019

Descrição

O objetivo desse trabalho é a implementação de um ray tracer simples.

Seu programa deverá ler uma entrada textual contendo uma descrição da cena a ser apresentada, e deve produzir um arquivo com a imagem renderizada da cena.

Não poderão ser utilizados outros projetos como projeto base, no máximo pode-se utilizar uma biblioteca auxiliar para gravar as imagens finais.

Sintaxe Do Programa

```
tp2 infile.txt outfile.ppm [X Y]
```

infile.txt é um arquivo de descrição textual de uma cena 3D, conforme detalhado abaixo.

outfile.ppm é o arquivo de saída, que contem a imagem renderizada da cena, podendo estar no formato PPM, também descrito abaixo, ou em qualquer outro formato padrão que você se sinta confortável para gerar (GIF, JPEG, BMP).

Os parâmetros **X** e **Y** são opcionais, e representam as dimensões da imagem a ser gerada na saída. Caso não estejam especificados, a imagem gerada deve ter o tamanho 400x300.

Testes Durante o Desenvolvimento

Você receberá algumas cenas de teste para depurar o seu programa durante o desenvolvimento.

Você pode criar novos arquivos que achar interessantes.

Entrega

Você deverá incluir um arquivo README.TXT contendo todos os detalhes da sua implementação.

Nele você deverá colocar as instruções necessárias para compilação e execução do seu código.

A entrega será pelo SGA e um pacote (arquivo ZIP ou tar.gz) com todos os arquivos necessários à compilação do seu código, além dos arquivos de descrição das cenas adicionais que você construir.

Formato da Entrada

A arquivo de entrada do programa contem a descrição de uma cena em 3D.

Cada objeto pertencente à cena deve ser descrito não apenas em termos de sua geometria, mas também suas propriedades para renderização, consistindo de sua cor, além das propriedades acabamento da superfície (ou as propriedades de reflexão de luz).

O formato vai ser amigável ao programa (e não necessariamente ao usuário), de forma a reduzir o esforço na leitura da especificação da cena, podendo, o programador concentrar na implementação do ray tracer propriamente.

Todos os pontos e vetores são descritos como coordenadas tridimensionais (x, y, z), em relação ao mesmo sistema de coordenadas do mundo.

Todas as cores são descritas como um vetor RGB, contendo valores reais (tipicamente entre 0 e 1, mas não necessariamente).

A Câmera

O arquivo de entrada tem um formato bem rígido e específico começando com 4 linhas contendo informação básica de câmera e perspectiva.

Os parâmetros são as coordenadas do olho, e o ponto do centro da cena.

Depois deverá conter um vetor apontando para "cima" e finalmente o campo de visão (em graus).

Não haverão os planos de corte próximo e distante.

O plano de projeção deverá estar a uma unidade de distância do olho, na direção do vetor de direção de visualização (centro – olho).

A razão de aspecto da cena deverá ser determinado pelos parâmetros que especificam as dimensões da imagem de saída (valor default de 400x300, se não especificado).

Abaixo apresentamos um exemplo do início do arquivo de entrada. Note que os comentários (começados com # e estendendo até o final da linha) não precisam ser tratados pelo programa.

```
        # COMETÁRIOS
1-10 5  # olho no ponto (1,-10, 5)
1 10 -3 # olhando na direção de (1, 10,-3)
0 0 1   # o vetor normal à cena (eixo Z)
20      # campo de visão de 20 graus
```

A partir desses parâmetros, deverá ser gerado o sistema de coordenadas da câmera (contendo a origem e os três vetores ortonormais) a partir do qual os raios serão lançados (na direção do centro de cada pixel da imagem).

Fontes de Luzes

O arquivo conterá uma linha contendo um inteiro NL, representado o número de fontes de luz presentes na cena.

Essa será seguida de NL linhas, uma para cada fonte de luz, contendo três triplas de valores reais, especificando, a posição da fonte (x, y, z), sua cor, dada pelas intensidades de suas componentes (R, G, B) e os componentes de atenuação (a, b, c), segundo a fórmula de atenuação com a distância $\boldsymbol{d}$ a partir da fonte de luz: $\mathbf{1/(a + bd + cd^2)}$.

As fontes de luz serão numeradas de 0 até NL - 1, sendo a fonte 0 sempre considerada a iluminação ambiente (sua localização, e parâmetros de atenuação, apesar de presentes na entrada, deverão ser ignorados pelo programa).

A seguir, um exemplo do arquivo de entrada:

```
2                           # duas fontes de luz
0 0 0 0.5 0.5 0.5 0 0 0     # luz ambiente branca
0 10 50 1.0 0.0 0.0 0 0.1 0 # luz vermelha na posição ( 0, 10, 50 )
```

Pigmentos

Haverá na entrada uma linha contendo um inteiro NP com o número de pigmentos presentes na cena, seguida de NP linhas, uma para cada pigmento.

Os pigmentos também são numerados entre 0 e NP-1.

Os pigmentos podem ser entendidos como uma função de mapeamento entre um ponto (x, y, z) no espaço da cena e uma cor em RGB.

Vários tipos de pigmentos podem ser incorporados.

Em particular, 2 tipos de pigmentos devem estar presentes (sólido e quadriculado), sendo que para cada um deles será especificado o seu tipo, seguido de parâmetros específicos para cada tipo.

Pigmentos sólidos consistem de uma cor sólida e seus parâmetros são a palavra chave "solid" seguida da cor em RGB.

Pigmentos quadriculados são especificados pela palavra chave "checker" seguida de duas cores (também em RGB), e um tamanho (número real) dos quadrados (cubos, no espaço 3D da cena) a serem gerados. Esse formato corresponde a uma textura procedural básica.

A seguir, um exemplo do arquivo de entrada, na sessão de pigmentos:

```
2                       # 2 tipos de pigmentos presentes
solid 0.0 0.4 0.0       # cor verde escura sólida
checker 1 0 0 0 0 1 2.0 # cubos de tamanho 2, vermelhos e azuis
```

Cor de Fundo Padrão

Deve existir uma cor padrão (vamos assumir como sendo cinza, RGB = (0.5, 0.5, 0.5)) que deve ser atribuída ao pixel, caso o raio correspondente não alcance objeto algum.

Acabamentos/Materiais dos Objetos

Na sequência, o arquivo de entrada contém a descrição dos diversos acabamentos de superfícies presentes na cena.

Seguindo o formato padrão, haverá uma linha contendo um inteiro NA com o número de acabamentos, seguida de NA linhas, contendo a especificação de cada um deles, numerados entre 0 e NA-1.

Esses acabamentos são especificados com 7 números reais, correspondendo aos 4 parâmetros de Phong, coeficiente de iluminação ambiente, difuso e especular, além do brilho (ns), seguido dos parâmetros de reflexão, transmissão, e o coeficiente de transmissão do interior do objeto (assuma que o meio externo é o ar, cujo coeficiente de transmissão é igual a 1).

A seguir, o exemplo do arquivo de entrada, com os parâmetros de acabamentos de superfícies:

```
2                          # 2 tipos de acabamentos presentes
0.3 0.1 1.0 500 0.9 0.0 0  # 0: altamente especular e reflectivo
0.0 0.7 0.0 50 0.0 0.5 1.5 # 1: difuso, parcialmente transparente
```

Objetos

Finalmente, o arquivo trás os objetos que compõem a cena.

Novamente haverá uma linha com o número de objetos NO, seguida de NO linhas contendo os objetos e sua especificação.

Para cada objeto, a sua linha de descrição começa com dois inteiros contendo o seu pigmento, e seu acabamento (entre 0 e NP, e 0 e NA, respectivamente), seguido do identificador do tipo do objeto, e seus parâmetros.

Dois tipos de objetos devem estar presentes: esferas e planos.

As esferas são identificadas pela palavra chave "sphere" seguido das coordenadas do centro e o raio da esfera.

Os planos são identificados pela palavra chave "plane", seguida da tupla (a, b, c, d), correspondendo aos coeficientes da equação do plano $\boldsymbol{ax + by + cz + d = 0}$. Para planos "transparentes" assuma que o interior do plano é dado pela inequação $\boldsymbol{ax + by + cz + d < 0}$.

A seguir, o exemplo da descrição de objetos no arquivo de entrada:

```
2                   # 2 objetos
0 0 sphere 2 0 -5 2 # esfera sólida de raio 2, centrada em (2, 0, -5)
1 1 plane 2 1 0 10  # plano quadriculado 2x + y + 10 = 0
```

Observação Final

Assuma, para fins de simplificação, que não haverão mais do que 20 fontes de luz, 50 pigmentos, 50 acabamentos de superfície e 200 objetos em uma única cena.

Format de Saida

É indicado o formato PPM por ser mais simples.

O mesmo pode ser visualizado ou transformado para outros formatos por um conjunto grande de ferramentas disponíveis livremente na internet (exemplo: Gimp).

O arquivo possui um cabeçalho de 3 linhas em formato texto, sendo que a primeira contém um identificador do formato do arquivo (que podem ser bitmap ou pixmap, binário ou texto, monocromáticos ou coloridos), a segunda contém as dimensões da imagem como dois inteiros X Y, e a terceira contendo o maior valor possível para a cor de um elemento (sendo sempre zero o menor valor).

Depois dessas três linhas, o arquivo é meramente uma sequencia de X*Y inteiros entre zero e o máximo indicado no cabeçalho, correspondendo às cores dos pixels, dispostos em ordem rowmajor(uma linha na frente da outra).

Dependendo do formato, esses inteiros podem estar em modo texto ou em binário (um byte por inteiro).

Podem haver também 1 byte ou 3 bytes para cada pixel da imagem.

O fomato PPM propriamente, usa três bytes por pixel, e esses são apresentados em binário.

Portanto, o arquivo pode ser lido diretamente (com um read) para uma região na memória com X*Y*3 bytes (após a leitura do cabeçalho).

A escrita do arquivo também é extremamente simples.

Mais detalhes podem ser encontrados na internet.

Implementação

Para projeção de sombra, assuma que objetos "transparentes" projetam sombra, ou seja, a luz só incidirá sobre um objeto, se ela for atiginda pelo raio saindo do objeto sendo renderizado antes de qualquer outro objeto.

Assuma ainda um número máximo de 10 iterações para reflexão e refração.

Depois desse limite, aplique a cor padrão (cinza) no pixel corresponde (agregando com os outros componentes encontrados).

Ponto Extra

Qualquer técnica de CG que você colocar adicionalmente no TP será considerada para pontuação extra.
Exemplos:

- Texturas definidas por gradientes: dois pontos que definem uma cor espacialmente, e os valores entre esses dois pontos é o resultado da interpolação dessas cores.
- Texturas definidas por imagens.
- Implementar outros tipos de objetos: além do plano e esfera: cones, cilindros, toroides, quádricas, etc... Pada cada nova definição de sólido, deve-se acrescentar uma entrada nova na especificação dos objetos.
- Implementar outros modelos de câmeras. O Tp tem a definição de uma câmera simples, mas é possível implementar câmeras com mais parâmetros como: distância focal, abertura, etc...
- Nevoa (FOG) é possível acrescentar o efeito de névoa à cor final de acordo com o comprimento do raio, sendo que essa cor deve ser misturada à cor final.
- Calcular sombras com umbras e penunbras através do lançamento de vários raios a partir de um ponto
- etc...