Ray Tracing

24 de setembro de 2010

1 Introdução

Ray tracing é um método para a geração de imagens realísticas por computador. Apesar da ideia simples e de implementação relativamente fácil, a construção de um programa eficiente e que trate de objetos complexos é bastante complicada.

Neste exercício o objetivo é fazer um trabalho intermediário, onde o desempenho continua sendo o principal fator, mas usaremos objetos de complexidade mediana. O trabalho poderá ser feito por grupos de até 3 pessoas.

Para entender o que é a técnica de *ray tracing*, imagine que iremos tirar uma fotografia virtual: existe um "mundo" com vários objetos geométricos[1], cada um com propriedades diferentes de interação com a luz [2] formando um cenário.

Um retângulo no espaço será a janela por onde olhamos este cenário e um outro ponto será a posição da vista ou da câmera.

A janela será mapeada em um arquivo de imagem (PNM). Cada pixel corresponde a um ponto da janela. Sua fica determinada pelas leis da ótica clássica, da seguinte forma. Considere o raio que atinge a vista passando por este ponto (pixel) e percorra o caminho inverso, calculando todas as reflexões, refrações e absorções, até que o raio se perca no infinito, fique com muito pouca intensidade ou atinja uma fonte de luz. A cor do raio deve ser composta de acordo.

Cada raio pode ser tratado independentemente, mas as dificuldades estão nos cálculos das interseções raio-objeto e no balanceamento de carga.

[1] Usaremos segmentos de planos, esferas, cones e cilindros

[2] Ex. cor, reflexividade, refração, absorção e transparência

Veja um tutorial em `http://fuzzyphoton.tripod.com/`. Note que o programa pode ser recursivo ou não, depende de como se deseja implementar.

2 O Mundo

O mundo, ou o cenário, será composto por uma coleção de objetos, cada um com uma especificação de material.

Para simplificar a implementação, a entrada será suposta consistente, isto é, não será necessário se preocupar com a correção dos dados. Por exemplo: se um sólido apresentar as faces com o lado "externo" apontando para dentro, assim ele será tratado pelo programa.

2.1 Objetos

Os tipos de objeto que podem aparecer no mundo são os seguintes:

- Triângulos (faces bi-dimensionais orientadas)
- Esferas
- Cilindros
- Cones

Cada objeto será comentado a seguir.

2.1.1 Triângulos

Triângulos são interessantes como base para composição de sólidos, pois são sempre planos (simplexos)[3]. Qualquer poliedro pode ser facilmente decomposto em uma coleção de triângulos. Desta forma, eles são excelente para descrever faces planas.

Na composição de sólidos, bem como para determinar qual o ângulo de refração, é muito importante saber qual é a face externa e qual a interna. Vamos convencionar que os triângulos serão sempre fornecidos com os vértices

[3]A menos que os vértices sejam colineares, mas neste caso não se trata de um triângulo

no sentido anti-horário por quem olha "de fora". Desta forma, para um triângulo $(\vec{p_1}, \vec{p_2}, \vec{p_3})$, a normal é dada por

$$\vec{n} = (\vec{p_2} - \vec{p_1}) \times (\vec{p_3} - \vec{p_2})$$

Dado o raio **r** e um triângulo $\mathbf{T} = (\vec{p_1}, \vec{p_2}, \vec{p_3})$, é (deveria ser) simples determinar o ponto de interseção entre o **r** e o plano de **T**. Se $\vec{v}$ for este ponto, o teste para saber se ele é interno ou não a **T** pode ser decomposto em três testes similares:

$$\begin{aligned} \vec{n} \cdot (\vec{p_2} - \vec{p_1}) \times (\vec{r} - \vec{p_1}) &> 0 \\ \vec{n} \cdot (\vec{p_3} - \vec{p_2}) \times (\vec{r} - \vec{p_2}) &> 0 \\ \vec{n} \cdot (\vec{p_1} - \vec{p_3}) \times (\vec{r} - \vec{p_3}) &> 0 \end{aligned}$$

onde $\vec{n}$ é a normal, $\cdot$ é o produto escalar e $\times$ é o produto vetorial. Geometricamente, isto significa que o ponto está sempre à esquerda de quem percorre as arestas no sentido anti-horário. Note que $\vec{p_j} - \vec{p_i}$ é a aresta orientada que vai de $\vec{p_i}$ a $\vec{p_j}$.

2.1.2 Esferas

Para saber se um ponto é interior a uma esfera ou não, basta medir sua distância ao centro e comparar com o raio. Achar o ponto de interseção entre o raio e a superfície da esfera, deve-se resolver uma equação quadrática simples. A normal ao ponto de interseção também é facilmente encontrada pela diferença entre a interseção e o centro da esfera. *Enough said.*

2.1.3 Cilindros

Muito parecido com esferas. A diferença é que agora deve ser testada a distância entre o raio e o eixo do cilindro. Para as "tampas" superior e inferior o tratamento é parecido com o dos triângulos, só que o teste de pertinência é mais fácil, correto?

2.1.4 Cones

Apenas ligeiramente mais difícil do que o cilindro. Feito um, o outro é uma pequena adaptação.

2.2 Materiais

Para os materiais, serão considerados apenas as características mais pertinentes a este problema: cor, refração, transparência, reflexão e luminosidade. Estes elementos serão usados para determinar completamente o raio de luz que atinge a janela.

Para tanto, é preciso primeiro caracterizar completamente o **raio de luz**. Ele é composto pelos seguintes elementos:

1. Direção, definida como uma semi-reta, ou uma reta orientada.
2. Fator de cor, representada por um tripleto de reais entre 0. e 1.. Este tripleto armazenará correção que deve ser aplicada na cor final.
3. A indicação do *pixel* correspondente a este raio.
4. Geração, um inteiro que é inicialmente 0 e que é incrementado a cada reflexão ou refração (veja abaixo).

Para a imagem final, a cor será representada por um tripleto, com valores entre 0 e 255 para cada elemento, na tradicional ordem *RGB*. A refração é representada por um número real (índice de refração). Os outros elementos serão representados por tripletos com componentes entre 0. e 1..

A *luminosidade* do material é a quantidade de luz que ele emite em cada componente. Uma fonte branca, por exemplo, tem luminosidade $(1., 1., 1.)$.

A *reflexão* indica, para cada componente, a quantidade de luz refletida. Um objeto azul tem reflexão $(0., 0., 1.)$, enquanto um cinza possui $(0.5, 0.5, 0.5)$, por exemplo.

A *refração* é o índice de refração do lado interno do material. Consideraremos o mesmo índice para todas as frequências de luz, para simplificar. Este índice será usado para determinar a nova direção do raio de luz ao atravessar uma superfície, segundo as regras da ótica clássica. Sabendo-se qual é o lado "de fora" e o lado "de dentro", é possível realizar o cálculo. O valor deve representar o índice de refração relativo entre o lado externo e interno. Suponha que a entrada é consistente ao montar os objetos.

A *transparência* representa a cor interna do material. Um material opaco tem transparência $(0., 0., 0.)$.

Quando um raio atinge uma superfície, as seguintes efeitos acontecem:

1. Um raio é criado pela reflexão, com um valor de *geração* incrementado em um, com direção ditada pela reflexão e com os componentes de cor multiplicados, um a um, pela correção correspondente.

2. Um raio é criado pela refração, de forma similar.

3. Se houver luminosidade, esta é corrigida pelo fator de cor do raio *incidente* e o resultado é imediatamente somado à cor do *pixel*.

Na entrada será fornecido um parâmetro chamado de *limiar de visão*. Um raio de luz cuja maior componente do fator de cor for inferior a este parâmetro será descartado do processamento. O mesmo acontece com raios que possuam geração maior do que o máximo previsto na entrada.

Em sala discutiremos os raios que atingem o infinito. Por *default*, serão descartados.

3 O programa

O programa receberá os seguintes parâmetros como entrada:

- Nome do arquivo de especificação do mundo, cujo formato está descrito na seção 4
- Nome do arquivo com a posição do ponto de vista e da janela, descrito a seguir.
- Nome do arquivo de saída
- Largura da imagem.
- Altura da imagem — o valor zero indica que a altura deve ser calculada de modo a manter a proporção da janela.
- Valor do *limiar de visão*. É um número entre 0. e 1..
- Número máximo de gerações de um raio. É um inteiro positivo.

A sintaxe de chamada será a seguinte:

programa descmundo.txt descvista.txt limiar gerações

Por exemplo:

```
rayt mundo.rt  vista.vs saida.ppm 800 600 0.002 10
```

Você poderá incluir parâmetros *opcionais* depois do último, se quiser.
O arquivo que descreve a vista é bem simples, em formato texto:

1. Na primeira linha estão as coordenadas do ponto de visão.
2. Na segunda linha estão as coordenadas do ponto inferior esquerdo da janela (referente à imagem final)
3. Na terceira e última linha estão as coordenadas do ponto superior direito da janela.

4 Formato do arquivo de descrição do mundo

O mundo é descrito por um arquivo texto, que consiste na lista dos objetos e seus materiais. Cada bloco corresponde a um elemento e possui a seguinte sintaxe, linha a linha:

1. Código do objeto:

   0 Triângulo

   1 Esfera

   2 Cilindro

   3 Cone

2. **Luminosidade**: 3 números reais
3. **Reflexão**: 3 números reais
4. **Transparência**: 3 números reais
5. **Refração**: 1 número real

As linhas seguintes dependem do tipo do objeto:

Triângulo Seguem-se 3 linhas, cada uma com as coordenadas dos vértices, apresentados em sentido anti-horário quando olhados pelo lado externo da face.

Esfera Uma linha com o raio e as coordenadas do centro

Cilindro Seguem-se 3 linhas: Na primeira estão as coordenadas do centro da face superior, na segunda as coordenadas do centro da face inferior e na terceira linha está o raio do cilindro.

Cone Similar ao cone, mas na terceira linha encontram-se os dois raios: superior e inferior, nesta ordem.

Exemplo:

```
1
0. 0. 0.
.3 .21 .92
.4 .5 .4
0.56
12.0 50. 20. 12.
0
.8 .2 .6
.1 .2 .3
.6 .6 .6
.98
12.1 3.4 -12.41
23.3  -12.  45.32
-9. 0.65 23.42
```

5 Construção da imagem

Em breve....

6 Regras

Vale todo tipo de otimização e paralelização, os programas serão excutados na mesma máquina (pegrande) e com parâmetros escolhidos por mim.

A eficiência relativa do programa conta um ponto na nota do EP e será calculada da seguinte forma: o(s) programa(s) mais rápidos recebe(m) 1 ponto, o mais lento recebe 0 (no bônus apenas), o restante recebe um valor proporcional à diferença. Para evitar uma concentração exagerada em 1, retirarei os tempos mais lentos até que a mediana chegue a 0,5. Posso fazer o

mesmo no outro sentido, caso apenas um programa consiga uma velocidade muito maior do que os outros, a meu critério.

Que a velocidade esteja com vocês.