UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E AUTOMAÇÃO INDUSTRIAL

Trabalho de Computação Gráfica

Traçado de Raio

Autores: **Augusto Valente**

Wallace Loos

Profa. Dra. Wu, Shin - Ting

Junho 2011

Resumo

O trabalho apresentado tem como objetivo explorar uma técnica de iluminação global, o traçado de raio. Quando queremos alcançar um nível de realismo em imagens criadas, temos que levar em consideração as interações que acontecem entre os objetos e não apenas as interações providas diretamente das fontes de luz. O realismo das imagens foi o ponto de partida para o estudo dessa técnica, que é amplamente usada em jogos, produções de filme, etc.

Lista de Figuras

Sumário

1 Indrodução

Na computação gráfica, a iluminação global é o termo dado para modelos que renderizam a visão de uma cena calculando a luz refletida, ou seja, não considera se somente a luz vindo diretamente da fonte, mas também aquelas que chegam indiretamente[5]. Modelos de iluminação global estão relacionados ao realismo das cenas geradas sinteticamente. Dois algoritmos de iluminação global (parcial) são propostos, traçado de raio e radiosidade. No trabalho abordaremos apenas o algoritmo de traçado de raio. Ambos simulam um subconjunto das interações globais, traçado de raio interações especulares, radiosidade interações difusas[5].

(a) (b) (c)

Fig. 1: Imagens reproduzidas utilizando a técnica de traçado de raio. (`http://en.wikipedia.org/wiki/Ray_tracing_(graphics)`)

A ideia do traçado de raio é a de traçar raios na cena e para cada objeto em que o raio intercepte, seja calculada a intersecção, e para o ponto interceptado seja gerado um raio refletido e outro refratado, dependendo da propriedade do material. Podemos adotar duas estratégias para implementação do traçado de raio: *forward e backward*. Na estratégia *forward* analisa se os raios que saem da fonte luz e calcula se as intersecções caso o raio intercepte algum objeto. A desvantagem em se usar essa estratégia está no fato de que da fonte de luz saem diversos raios e não estamos interessados em todos eles.

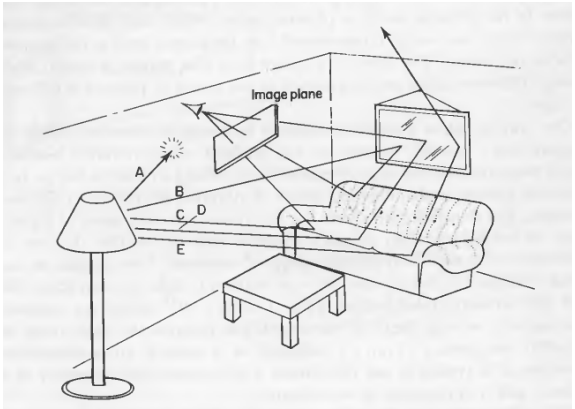


Fig. 2: Raios de luz sendo lançados de uma fonte de luz (estratégia *forward*)[2].

Estamos interessados somente nos raios que chegam ao observador, ou seja, apenas em um subconjunto de raios. A Fig. 2 , mostra alguns raios que saem do

abajur. Observamos que os raios A e E não chegam ao observador, assim não seria necessário calcular alguma intersecção. O gargalo do algoritmo de traçado de raios é exatamente o cálculo das intersecções. Adotando se essa estratégia calcularemos intersecções desnecessárias. Na estratégia *backward* percorremos o caminho inverso. Ao invés de saírmos da fonte de luz, saimos do próprio observador, pois se estamos vendo os objetos que compõe a cena, então existem raios de luz que incidiram naqueles objetos. Lançamos para cada *pixel* do plano de imagem um raio que para cada objeto em que ele interceptar calcularemos a intersecção e geraremos um raio refletido e outro refratado, dependendo da propriedade do material. Então analisaremos somente os raios que nos interessa. A diferença de uma estratégia da outra está no fato de percorrermos o raio a partir da fonte de luz e a partir do observador.

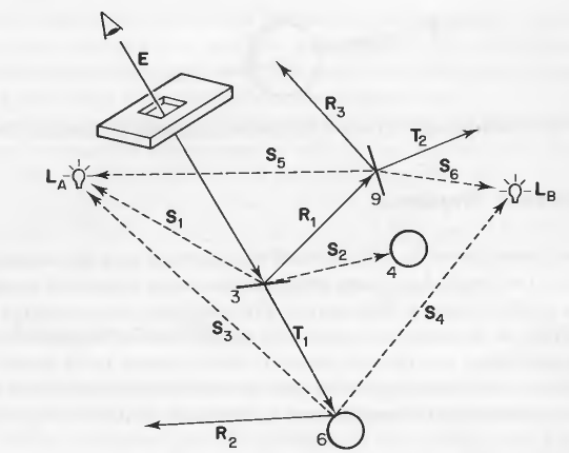


Fig. 3: Interações dos raios de luz[2].

A Fig. 3 mostra um raio que é lançado do observador. O raio intercepta uma superfície transparente gerando dois novos raios R_1 e T_1, que são respectivamente o raio refletido e o raio transmitido ou refratado. Da superfície é lançado um raio S_2 chamado de raio de sombra, esse raio verifica a existência de sombra. A cada ponto atingido os mesmo cálculos são realizados, isso nos leva a uma simples implementação recursiva. O processo geralmente é visualizado como uma árvore onde cada nó é um ponto atingido da superfície[2]. A recursão pode terminar de acordo com o número de critérios

- Termina quando intercepta uma superfície difusa.
- Termina quando atinge uma certa profundidade.
- Termina quando a energia do raio ficar abaixo de um limiar pré-definido.

Em 1980, Turner Whitted propôs um algoritmo de iluminação global parcial que combina

- Visibilidade
- Tonalização devido a iluminação direta

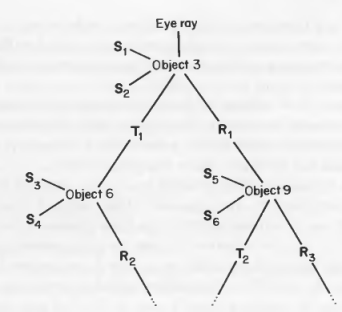


Fig. 4: Processo de interações em árvore[2].

- Efeitos de interações especulares globais tais como a reflexão dos objetos e refração da luz através de objetos transparêntes
- Cálculo da sombra

1.1 Componentes do Modelo de Iluminação

Uma questão crucial para compor uma imagem é determinar a cor correta de cada *pixel*[2]. Mas como encontrar os raios e suas cores que influenciam na determinação da cor do *pixel*? Podemos pensar no raio de luz como um caminho reto, que é percorrido por partículas de luz, chamadas *fótons*. A energia carregada pelo *fóton* é percebido por nós através de nossas células receptoras, a cor percebida está relacionada com a energia dos *fótons*. Para cada ponto **P** atingido pelo raio, em geral, é gerado dois raios: raio refletido e o transmitido ou refratado, também é calculado um modelo de reflexão local [5]. O modelo de reflexão local é necessário pelo fato de existir uma iluminação direta que atinge o ponto. Assim em cada ponto a intensidade da luz consiste em três componentes

- Uma componente local,
- Uma contribuição do raio refletido,
- Uma contribuição do raio transmitido (refratado).

Para cada ponto **P** que é atingido com um raio, serão considerados duas componentes principais, uma componente local e outra global.

$$\begin{aligned} I(P) &= I_{local}(P) + I_{global}(P) \\ &= I_{local}(P) + k_{rg}(P_r) + k_{tg}I(P_t) \end{aligned} \tag{1}$$

onde

- P: é o ponto atingido,
- P_r: o raio refletido no ponto P,

- P_t: o raio transmitido (refratado) no ponto P
- k_{rg}: é o coeficiente de reflexão global,
- k_{tg}: é o coeficiente de transmissão global.

Para o modelo de iluminação local, geralmente é usado o modelo de Phong, conforme[5]. Para o modelo global calculamos P_r e P_t.

1.2 Sombra

As sombras são incluídas facilmente no algoritmo de traçado de raio. Para determinar a iluminação no ponto **P**, perguntamos se algum raio de luz viaja até o ponto **P**. Para responder a essa pergunta lançamos um raio de iluminação em direção a fonte de luz, se o raio chega a fonte de luz, então aquela fonte está iluminando diretamente a superfície, caso o raio lançado não chegue a fonte de luz então algum objeto está bloqueando o raio, assim a fonte de luz não está iluminando diretamente aquela superfície, então haverá formação de sombra.

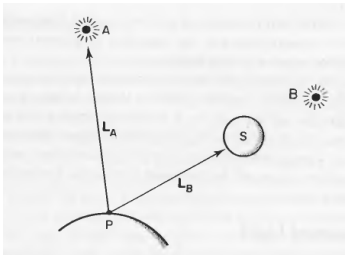


Fig. 5: Raio de sombra ou de iluminação sendo lançado da superfície. O raio L_A chega diretamente a fonte de luz A, o raio L_B é bloqueado por um objeto, e não chega a fonte de luz B, ocorrendo a formação de sombra[2].

1.3 Visibilidade

A visibilidade está incluído no algoritmo de traçado de raio [5]. Para cada raio é realizado um teste onde é verificado se o raio intercepta algum objeto.

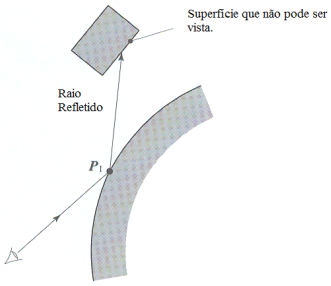


Fig. 6: Representação de uma superfície que fica invisível ao observador[5].

O resultado desse teste será uma lista de objetos que o raio intercepta. Os objetos que são interceptados e que estão fora do raio de visão do observador, não terão suas intersecções calculadas, pois o observador não estará vendo tal superfície.

2 Noções de Implementações

O coração de qualquer algoritmo de traçado de raio é o conjunto de rotinas que realizam as intersecções[4].

2.1 Cálculo de raios

Dados um observador posicionado no ponto **O** e um ponto no plano de imagem com coordenadas espaciais P_{pixel}, a direção do raio que deve ser traçado é $R_d = P_{pixel}$ - **O**, e sua origem será em **O**. Este raio interceptará um objeto na cena no ponto P_{obj}. Dada uma fonte de luz em P_{luz}, o raio de iluminação (vide seção 1.2) terá direção $L_d = P_{luz}$ - P_{obj} e origem em P_{obj}.

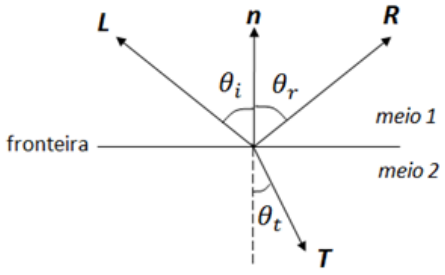


Fig. 7: Reflexão e refração perfeitas na interação de um raio com uma superfície.

Quando um raio incide em algum objeto ou em alguma superfície podem ser gerados dois outros raios, um refletido e outro refratado, de acordo com as propriedades do material em questão. Cada raio contribui na cor em que será atribuída ao pixel (vide seção 1.1). A obtenção da direção do raio refletido é ilustrada na Fig. 7, onde os raios incidente (**L**) e refletido (**R**) fazem o mesmo ângulo com a normal à superfície n, ou seja, $\theta_i = \theta_r$. A direção **R** do raio refletido é dada por $\mathbf{R} = 2(\mathbf{L}.\mathbf{n})\mathbf{n}$ - **L**. Nota-se que o sentido de **L** é da superfície para o observador, ou seja, $\mathbf{L} = -R_d$.

A direção do raio refratado depende dos índices de refração dos meios envolvidos e é calculado a partir da lei de Snell como

$$\mathbf{T} = \frac{\mathbf{L}}{\eta} - \left(\cos\theta_t - \frac{1}{\eta}\mathbf{L}\cdot\mathbf{n}\right)\mathbf{n} \tag{2}$$

onde η é o índice de refração relativo entre os meios e

$$\cos\theta_t = \left(1 - \frac{1}{\eta^2}\left(1 - (\mathbf{L}\cdot\mathbf{n})^2\right)\right)^{1/2} \tag{3}$$

As grandezas envolvidas neste cálculo são também ilustradas na Fig. 7. Nota-se que, caso o termo dentro da raiz na expressão de $\cos(\theta_t)$ seja negativo, não ocorre refração, sendo este o fenômeno chamado de reflexão total.

Gerar raios apenas nestas direções dadas pela reflexão e refração perfeitas consiste em uma simplificação das interações que ocorrem na realidade, na qual vários raios são refletidos ou refratados em torno dessas direções. Esta simplificação garante que o problema seja tratável computacionalmente, mas pode gerar artefatos ou um aspecto não realista na imagem final.

2.2 Intersecções de raios

No trabalho consideraremos algumas primitivas, como o plano e a esfera, devida a simples manipulação algébrica. Mas isso não quer dizer que o algoritmo de traçado de raio se restrinja a apenas essas primitivas.

2.2.1 Intersecção com Esfera

A esfera é uma das primitivas mais usadas no traçado de raio [4]. Para realizarmos a intersecção com a esfera definimos o raio como:

$$R_o = [X_o \quad Y_o \quad Z_o]$$
$$R_d = [X_d \quad Y_d \quad Z_d]$$

onde R_o é a origem do raio e R_d é a direção do raio. O raio é definido por um conjunto de pontos sobre a reta

$$R(t) = R_o + R_d t \tag{4}$$

Pontos sobre a reta onde $t < 0$ estão atrás do raio de origem. A equação 4 é a forma paramétrica da reta ou explícita. Isso significa que todos os pontos podem ser gerados variando o valor de t. A esfera é definida como

$$S_c = [X_c \quad Y_c \quad Z_c]$$

$$(X_s - X_c)^2 + (Y_s - Y_c)^2 + (Z_s - Z_c)^2 = S_r^2 \tag{5}$$

onde S_c é o centro da esfera e S_r é o raio da esfera e $[X_s \; Y_s \; Z_s]$ são os conjuntos de pontos da superfície. A superfície da esfera é expressada com uma equação implícita. Para resolver o problema da intersecção, a equação do raio 4 é substituída na equação da esfera 5

$$\begin{aligned} X &= X_o + X_d t \\ Y &= Y_o + Y_d t \\ Z &= Z_o + Z_d t \end{aligned} \tag{6}$$

Substituindo esses conjuntos de equações na equação da esfera nos temos

$$(X_o + X_d t - X_c)^2 + (Y_o + Y_d t - Y_c)^2 + (Z_o + Z_d t - Z_c)^2 = S_r^2 \tag{7}$$

em termos de t temos

$$At^2 + Bt + C = 0$$

onde

$$\begin{aligned} A &= X^2 d + Y^2 d + Z^2 d = 1 \\ B &= 2(X_d(X_o - X_c) + Y_d(Y_o - Y_c) + Z_d(Z_o - Z_c)) \\ C &= (X_o - X_c)^2 + (Y_o - Y_c)^2 + (Z_o - Z_c)^2 - S_r^2 \end{aligned}$$

$$t_0 = \frac{-B - \sqrt{(B^2 - 4C)}}{2} \qquad t_1 = \frac{-B + \sqrt{(B^2 - 4C)}}{2}$$

Quando o discriminante for negativo, a reta não intercepta a esfera. Uma vez que t é encontrado, o ponto de intersecção é:

$$r_i = [x_i \quad y_i \quad z_i] = [X_o + X_d t \quad Y_o + Y_d t \quad Z_o + Z_d t]$$

O vetor normal a superfície é definido como

$$r_n = \left[\frac{(x_i - X_c)}{S_r} \quad \frac{(y_i - Y_c)}{S_r} \quad \frac{(z_i - Z_c)}{S_r} \right]$$

2.2.2 Intersecção com o Plano

Para realizarmos a intersecção com o plano definimos o raio novamente como:

$$\begin{aligned} R_o &= [X_o \quad Y_o \quad Z_o] \\ R_d &= [X_d \quad Y_d \quad Z_d] \end{aligned}$$

onde R_o é a origem do raio e R_d é a direção do raio. O raio é definido por um conjunto de pontos sobre a reta

$$R(t) = R_o + R_d t \tag{8}$$

Defina o plano em termos de [A B C D]

$$Ax + By + Cz + D = 0$$

O vetor normal ao plano é definido como

$$P_n = [A \quad B \quad C]$$

Substituindo 7 em 9 temos

$$A(X_o + X_d t) + B(Y_o + Y_d t) + C(Z_o + Z_d t) + D = 0$$

resolvendo em função de t

$$t = -\frac{(AX_o + BY_o + CZ_o + D)}{AX_d + BY_d + CZ_d}$$

em notação vetorial

$$t = -\frac{P_n R_o + D}{P_n R_d} \tag{9}$$

3 Técnicas Avançadas

3.1 Otimização e Eficiência

A tarefa de otimizar e aumentar a eficiência do traçado de raio pode ser dividida em três estratégias distintas: (1) reduzir o custo médio intersecções do raio com o ambiente, (2) reduzir o número total de raios que interceptam o ambiente, e (3) realocar o raio individual por uma entidade mais genérica[2].

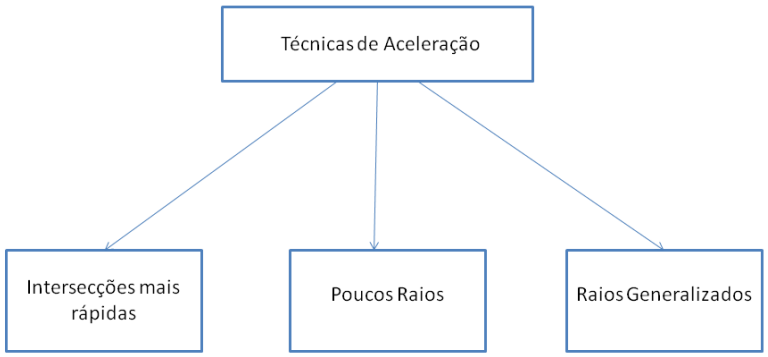


Fig. 8: Estratégias para otimizar o algoritmo traçado de raio.

A primeira estratégia consiste em utilizar eficientes algoritmo de intersecções para especificados objetos, além disso diminuir o número de testes de intersecções. Para isso podemos usar um volume limitante, esse volume contém o objeto e permite simplificar a checagem da intersecção do raio. A segunda estratégia nos permite reduzir o número de raios que interceptará o ambiente. Assim podemos estabelecer o controle de profundidade da árvore do processo do algoritmo de traçado de raio (Fig. 4), ou seja, estabelecemos antes do algoritmo começar uma profundidade máxima que a árvore pode alcançar. A terceira estratégia realoca o conceito familiar de raio por uma entidade mais genérica, por exemplo, cones ou feixes. A ideia do algoritmo é mantida[2].

3.2 Tratamento do Aliasing

Computadores são dispositivos discretos que representam sinais com um número finito de elementos. Imagens, por exemplo, são discretizadas espacialmente em elementos denominados *pixels*, que, por sua vez, possuem um conjunto discreto de possibilidades de intensidade (cor). O processo de discretizar um sinal contínuo é chamado de amostragem. Essa representação aproximada da realidade contínua pode gerar artefatos, e este processo é denominado *aliasing* (do inglês *alias*, disfarce, nome falso). Em imagens, o artefato mais óbvio é chamado *jaggie* (do inglês *jagged*, pontudo), que consiste no efeito serrilhado em regiões de bordas e transição, ilustrado na Fig. 9. Também são artefatos comuns a aparição de padrões inexistentes (padrões de Moiré) em regiões com detalhes finos e textura complexa e de bandas

de cores no lugar de dégradés suaves, amos exemplificados na Fig. 10. Objetos pequenos podem também ser totalmente excluídos.

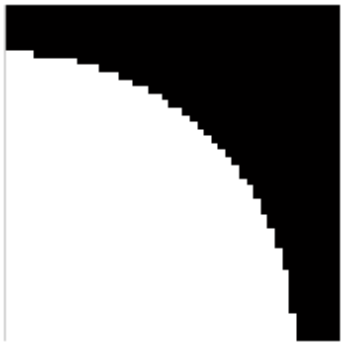
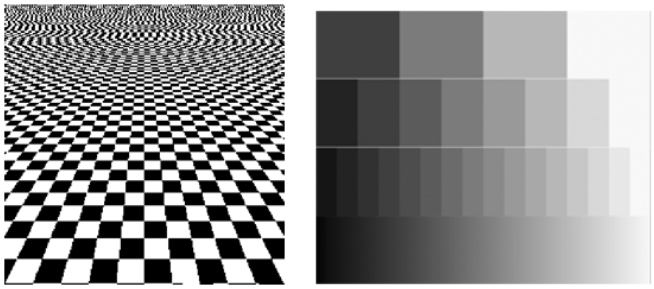


Fig. 9: Efeito serrilhado ou *jaggie*.



(a) (b)

Fig. 10: (a)Padrões de Moiré (ao fundo) e (b) transformação de degradé em bandas de cores.

O processo de amostragem é intrínseco à técnica de traçado de raio, pois um conjunto limitado de raios infinitesimalmente finos é traçado e a radiação é calculada apenas nos pontos por eles interceptados. Portanto, na maioria dos casos, o *aliasing* não pode ser totalmente eliminado usando-se a técnica clássica de traçado de raio, pois em geral haverá detalhes da cena que não serão captados pelo conjunto finito de raios traçados.

As técnicas mais comuns de tratamento de *aliasing* no traçado de raio tradicional são baseadas na mesma ideia: traçam-se vários raios para apenas um pixel[4]. Dessa maneira, espera-se que aquele elemento da imagem seja uma representação mais fiel da região correspondente a ele no espaço. Essas técnicas são tratadas nesse contexto como superamostragem. As diferentes organizações dos raios dentro do pixel acarretam em resultados diferentes na imagem. A maneira mais intuitiva é espaçá-los regularmente como na Fig. 11, que representa apenas um pixel. Os pontos representam por onde os raios traçados passarão. Na Fig. 11(b), a porção da cena idealmente retratada no pixel é sobreposta à grade uniforme. Na Fig. 11(c) é mostrada a média do resultado da tonalização de cada raio, que será o valor atribuído ao pixel. Nota-se que na cena há apenas duas cores, amarelo e cinza, mas o resultado é uma cor intermediária. A tendência é que a borda do objeto amarelo possua transições menos abruptas em relação ao fundo cinza, e sua representação seja mais agradável para um observador. Esta técnica pode acarretar na formação de padrões indesejados devido à uniformidade espacial da amostragem.

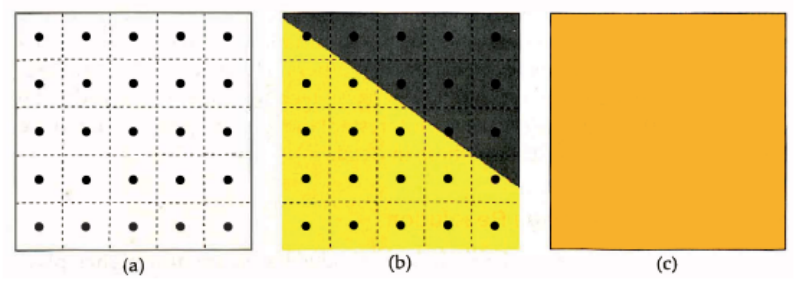


Fig. 11: Superamostragem uniforme: a) subdivisão do pixel em um *grid*, com um raio por célula, b) sobreposição da cena no *grid* e c) cor resultante do *pixel*.

Uma alternativa é distribuir os raios aleatoriamente no interior do pixel, o que substitui os padrões indesejados por ruído nas imagens. Isso pode ser útil pois a visão humana detecta melhor artefatos de *aliasing* do que ruído[2]. Portanto, uma imagem com ruído pode induzir uma percepção subjetiva mais positiva do que uma imagem com *aliasing*, embora o ruído não represente informação adicional verdadeira sobre a cena. Na distribuição puramente aleatória há a possibilidade de os raios se concentrarem em determinada região do pixel e não coletarem informação espacial suficiente. Uma solução de compromisso é dividir o pixel em um grid, como no caso uniforme, mas posicionar os raios aleatoriamente dentro desses *grids*. Essa técnica é denominada *jittered sampling*, e uma possível distribuição de raios um pixel é ilustrada na Fig. 12. Os raios podem ainda estar sujeitos a um viés, mas o efeito não será tão severo como na amostragem puramente aleatória.

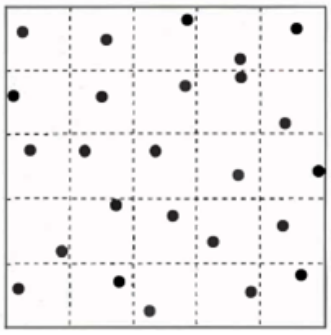


Fig. 12: Distribuição de raios em um pixel utilizando-se *jittered sampling*.

Técnicas mais avançadas consistem em adaptar o espaçamento do *grid* de amostragem ao longo da imagem de acordo com a necessidade. Por exemplo, regiões de borda e transição necessitam de *grids* finos para melhorar sua representação, enquanto que regiões sem variações abruptas podem ser bem representadas com menos samples. Dessa maneira, pode-se alocar o processamento de maneira inteligente onde ele for mais necessário. Além das possibilidades na distribuição de raios em um pixel, há também diferentes maneiras de se combinar esses raios em uma cor final. Além da média simples empregada anteriormente, pode-se ponderar os raios de acordo com sua posição, privilegiando aqueles mais próximos ao centro. Isso equivale a realizar uma filtragem 2D com diferentes máscaras. Caso o peso decline linearmente a partir do centro, tem-se uma máscara em formato de tenda. Caso

decline exponencialmente, tem-se o conhecido formato gaussiano. Pode-se também considerar raios fora da área do pixel, uma vez que o peso a eles atribuído será pequeno. As técnicas de superamostragem melhoram consideravelmente o aspecto da imagem final e são facilmente implementadas como extensão ao código de um traço de raio básico. Sua maior desvantagem é o grande aumento no tempo de renderização, dada a maior quantidade de raios a serem calculados.

3.3 Técnicas Estocásticas

3.3.1 Distributive Ray Tracing

A intensidade de um pixel na tela é determinada por uma função analítica que envolve várias integrais aninhadas: integrais ao longo do tempo, do espaço coberto pelo pixel, da área da lente por onde passam os raios, bem como uma integral do produto da refletância pela luminosidade sobre um hemisfério de reflexão e uma integral do produto da transmitância pela luminosidade sobre um hemisfério de transmitância. O cálculo dessas integrais é essencial para gerar efeitos como penumbras, reflexões borradas, translucidez, profundidade de campo e desfoques de movimento. Porém, seu cálculo direto é inviável computacionalmente, e diversas simplificações são feitas dependendo da técnica de renderização utilizada. Por exemplo, a equação de intensidade de reflexão em função da luminosidade L e refletância R para dados hemisférios de incidência (ϕ_i, θ_r) e de reflexão (ϕ_i, θ_r) é dada por:

$$I(\phi_r, \theta_r) = \int_{\phi_r} \int_{\theta_r} L(\phi_i, \theta_i) R(\phi_i, \theta_i, \phi_r, \theta_r) d\phi_i d\theta_i \tag{10}$$

Assumindo-se L como uma função delta não-nula apenas na direção das fontes de iluminação (pontuais), a integral pode ser tratada como uma soma discreta. Isso causa sombras agudas, sem penumbra. Assumindo-se que as direções que não correspondam a direções de fontes de luz podem ser agrupadas em uma fonte de luz ambiente, L passa a ser constante nessa região e a integral de R pode ser substituída por uma refletância média (ambiente). Assumindo-se que R é uma função delta não-nula apenas na direção de reflexão, obtêm-se reflexões agudas (sem *gloss*). Uma alternativa é utilizar técnicas estocásticas para avaliá-la. No caso do traçado de raio, utilizando-se integração de Monte Carlo, distribuem-se pontos de amostragem (os raios) em cada dimensão de integração necessária. Essa combinação de técnicas é denominada traçado de raio distribuído (*distributive ray tracing*) ou também traçado de raio estocástico (*stochastic ray tracing*)[2]. Como tratado na seção anterior, a amostragem puramente aleatória pode levar ao acúmulo das amostras em determinadas regiões, ignorando outras. Em geral, procura-se realizar uma técnica de *jitter* para simular alguma distribuição que tenha características desejáveis, como a de Poisson, por exemplo.

3.3.2 Efeitos Especiais

Utilizando-se a técnica estocástica da seção anterior, é possível modelar vários efeitos realistas.

- **Gloss**

  As reflexões tratadas no traçado de raio básico e na computação gráfica em geral são semelhantes às de espelhos, mas, na realidade, as reflexões são borradas e imperfeitas. Um cálculo analítico dessas reflexões requer uma integração da refletância sobre um ângulo sólido. As reflexões de espelhos são determinadas refletindo-se raios na direção de reflexão ideal. Utilizando-se o traçado distribuído, pode-se gerar raios em direções próximas à ideal, e combinar os resultados.

- **Translucidez**

  O efeito incluído nos algoritmos mais básicos em geral é a transparência, no qual as imagens vistas através do objeto são nítidas. Em objetos translúcidos, a imagem vista através deles não é totalmente distinta. É um problema análogo ao do *gloss*. Enquanto o último requeria uma integração da luz refletida, este requer uma integração da luz refratada (transmitida). A luz transmitida é descrita por uma equação similar à da seção anterior para a intensidade de reflexão, somente substituindo a refletância R pela transmitância T, e integrando-se ao longo do hemisfério por trás da superfície. Utilizando-se o traçado distribuído, a translucidez pode ser aproximada traçando-se raios em direções próximas ao do raio refratado ideal e combinando-se os resultados.

- **Penumbra**

  Penumbras ocorrem quando uma fonte de luz encontra-se parcialmente oclusa. A intensidade refletida em razão de tal fonte é proporcional ao ângulo sólido de sua porção visível. As sombras no traçado de raio básico são calculadas traçando-se um raio da superfície para a fonte de luz pontual. Similarmente nos efeitos anteriores, a penumbra pode ser obtida ao traçarem-se raios em direções aproximadas ao de sombra principal.

- **Profundidade de Campo (Depth of Field)**

  As câmeras reais e o olho humano possuem lentes com espessura finita, portanto as imagens geradas possuem profundidade de campo finita, i.e., a região de profundidade que se encontra em foco é uma faixa limitada. Cada ponto da cena aparece como um círculo no plano de imagem, e quanto maior o círculo mais desfocado está o ponto. Ao contrário, no traçado de raio clássico, assume-se o modelo de câmera *pinhole*, e todos os objetos são captados em foco, i.e., todos os pontos do espaço aparecem como um ponto no plano de imagem. Pode-se obter um efeito similar utilizando-se as técnicas estocásticas ao distribuir-se os pontos de origem dos raios que passam por um pixel como se estivessem sobre uma lente finita e combinando os raios resultantes.

- **Desfoque de movimento**

  O desfoque de movimento pode ser levado em conta ao distribuir-se os raios ao longo de pontos distintos no tempo. O ato de desfocar uma imagem no tempo incorre em diversos entraves como oclusões, desfoque de sombras e especularidades e outros. Métodos analíticos seriam inviáveis e a simples amostragem no tempo apresenta-se como melhor solução para este problema.

3.3.3 Roleta Russa

O cálculo sucessivo de reflexões no traçado de raio é em geral calculado com funções recursivas e necessita de um critério de parada. Em geral, fixa-se uma profundidade máxima de recursão e um limiar para a energia contida naquele raio, que sofre um decréscimo a cada interação. Em ambos os casos, o critério de parada é constante para todos os raios, e isso pode levar a artefatos devido ao viés introduzido na geração da imagem. Alguns trajetos longos de raios são importantes para certas características da cena e não podem ser descartados. A técnica de Roleta Russa trata o problema de manter o comprimento dos traçados em um limite razoável, mas ainda trabalha com a possibilidade de explorar trajetos de tamanhos arbitrários [3].

A ideia da Roleta Russa pode ser explicada através de um exemplo: suponha que se deseja calcular um determinado valor V. O cálculo de V pode ser computacionalmente complexo (e.g., requerendo a análise de integrais), portanto introduz-se uma variável aleatória r com distribuição uniforme no intervalo [0,1]. Caso r seja maior que um limiar $\alpha \in [0,1]$, prossegue-se no cálculo de V. Caso contrário, aborta-se o cálculo de V e assume-se $V = 0$. Tem-se, portanto, um experimento aleatório com valor esperado (1-α) V. Dividindo-se este valor por (1-α), obtém um estimador não tendencioso de V.

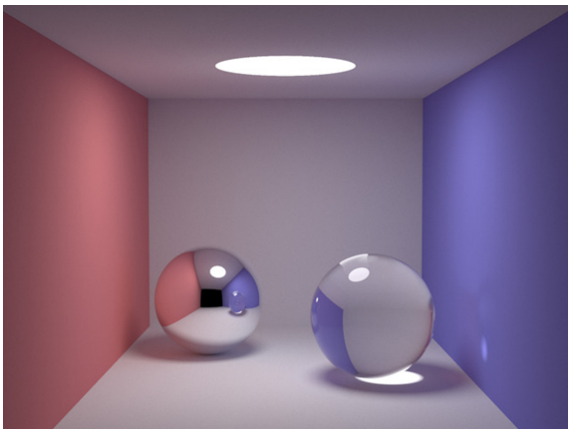


Fig. 13: Cena da caixa de Cornell gerada com um path tracer utilizando roleta russa.

Caso V requeira avaliações recursivas, como é o caso das reflexões e refrações no traçado de raio, pode-se utilizar este mecanismo para abortar a recursão. O parâmetro α é denominado probabilidade de absorção. Caso α seja pequeno, a recursão continuará por muitas vezes, e o valor final será mais preciso. Caso α seja

grande, a recursão acabará antes e o estimador apresentará maior variância. Em geral, o valor de 1-α é a refletância do material da superfície. Com isso, superfícies mais escuras tendem a absorver o raio mais facilmente, enquanto superfícies mais claras tendem a refletir o raio. Esta técnica é comumente utilizada em uma variação do traçado de raio denominada *path tracing*. O *path tracing* consiste em seguir o traçado de raio até que se atinja uma fonte de iluminação, sendo que as direções dos novos raios são geradas aleatoriamente no hemisfério de reflexão (ou refração) da superfície do objeto. Vários trajetos de raio serão perdidos pois não alcançarão uma fonte de luz, fazendo com que o método seja bastante ineficiente. O realismo das imagens obtidas, porém, é bastante superior a um traço de raio convencionais, como visto na Fig. 13.

4 Pov Ray

O *Pov Ray*[1] cria cenas foto realistas usando o traçado de raio [1]. A partir de um arquivo texto que contém informações descrevendo os objetos e a iluminação na cena, é gerada a imagem a partir do campo de visão da câmera especificado no arquivo texto. Desenvolveremos a seguir um simples exemplo apresentado em[1]. Primeiro devemos informar ao *Pov Ray* onde está nossa câmera e para onde ela está olhando. O sistema de coordenadas do *Pov Ray* possui o eixo y apontado para cima o eixo x para direita e o eixo z entrando na tela[1].

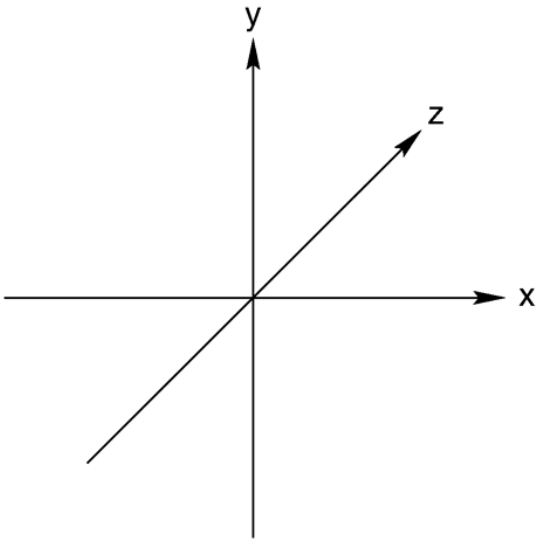


Fig. 14: Sistema de coordenadas do *Pov Ray*. Esse tipo de sistema de coordenadas é chamada de sistema de coordenadas da mão esquerda

4.1 Adicionando Arquivos Padrões

```
#include "colors.inc"
#include "stones.inc"
```

A primeira declaração inclui as cores. A segunda declaração inclui uma coleção de texturas de pedras. Além dessas existem outras, como:

```
#include "textures.inc"
#include "shapes.inc"
#include "glass.inc"
#include "metals.inc"
#include "woods.inc"
```

[1] Persistence of Vision Ray Tracer

elas pré definem as texturas da cena.

4.2 Adicionando uma Câmera

A declaração da câmera descreve sua posição e como a câmera irá ver a cena. Damos as três coordenadas para indicar sua posição e informamos para onde a câmera estará apontada. Nós descrevemos as coordenadas usando um vetor com três componentes

```
camera {
    location <0, 2, -9>
    look_at  <0, 1,  2>
  }
```

A parte da cena que a câmera vai ver é definido pelo comando *look_at<0, 1, 2>*. Esse comando aponta a câmera para as coordenadas (0, 1, 2).

4.3 Descrevendo um Objeto

As cenas no *Pov Ray* são descritas usando uma linguagem de texto especial chamada de linguagem de descrição de cena[1]. Um objeto é descrito em nossa cena da seguinte forma

```
sphere {
    <0, 1, 2>, 2
    texture {
      pigment { color Yellow }
    }
  }
```

O primeiro vetor especifica o centro da esfera. Nesse exemplo a coordenada x = 0, y = 1 e z = 2. Logo em seguida definimos o raio, que nesse caso foi igual a 2.

4.4 Adicionando Textura

Depois que definimos a localização e o tamanho da esfera, precisamos descrever a aparência da superfície. A declaração da textura define esses parâmetros. Texturas são blocos que descrevem a cor e as propriedades do objeto. Nesse exemplo iremos especificar somente a cor. Muitos tipos de cores são obtidos usando a declaração *pigment*. A cor chave especifica que o objeto inteiro terá essa cor. Os valores das cores estão entre 0 e 1. Qualquer componente não especificada será considerada 0.

```
color rgb <1.0, 0.8, 0.8>
```

4.5 Definindo uma Fonte de Luz

Precisamos de mais um detalhe a mais em nossa cena, uma fonte de luz. Adicionamos uma fonte de luz usando

```
light_source { <3, 1, -5> color White}
```

especificamos as coordenadas da fonte de luz e sua cor. Agora nosso exemplo está completo

```
#include "colors.inc"
  background { color White }
  camera {
    location <0, 2, -9>
    look_at  <0, 1,  2>
  }
  sphere {
    <0, 1, 2>, 2
    texture {
      pigment { color Blue }

    }
  }
  light_source { <3, 1, -5> color White}
```

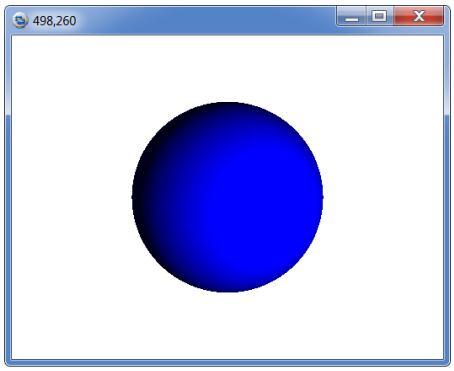
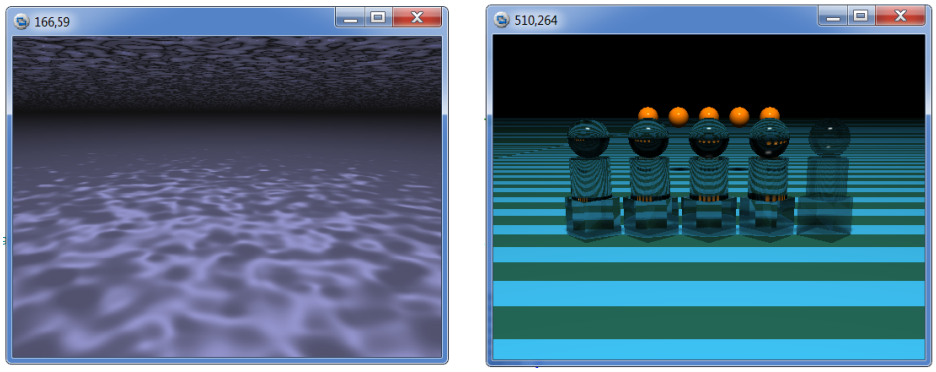


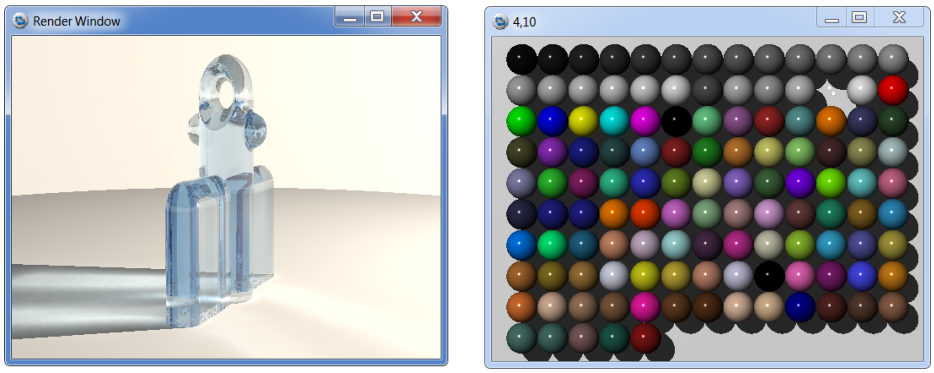
Fig. 15: Exemplo gerado.

Alguns exemplos foram gerados utilizando o programa *Pov Ray*.



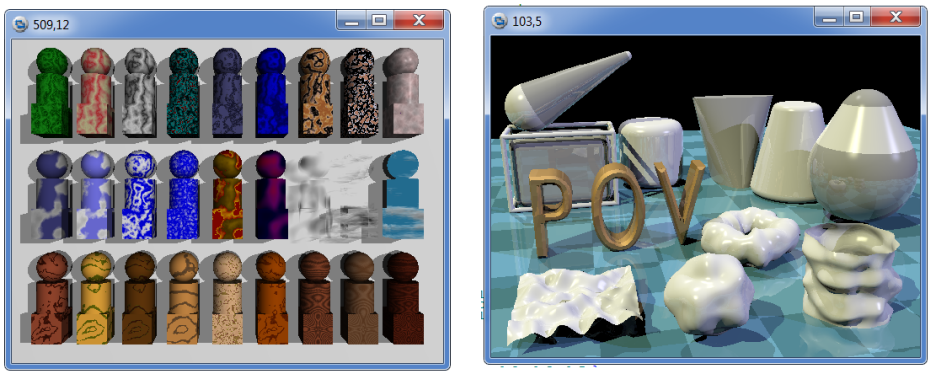
(a) (b)

Fig. 16: Imagens reproduzidas utilizando a técnica de traçado de raio[1].



(a) (b)

Fig. 17: Imagens reproduzidas utilizando a técnica de traçado de raio[1].



(a) (b)

Fig. 18: Imagens reproduzidas utilizando a técnica de traçado de raio[1].

Referências

[1] *Tutorial and Reference Material - Pov Ray.*

[2] Andrew S. Glassner. *An Introduction to Ray Tracing.* 1989.

[3] Philippe Bekaert Philip Dutré, Kavita Bala. *Advanced Global Illumination.*

[4] Kevin Suffern. *Ray Tracing from the Ground Up.* 2007.

[5] Allan Watt. *3D Computer Graphics.* Terceira edition, 2000.