COPPE/UFRJ

UMA COMPARAÇÃO DE MODELOS DE BRDFS PARA PHOTON MAPPING

Alberto Barbosa Júnior

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientadores: Antonio Alberto Fernandes de Oliveira
Ricardo Guerra Marroquim

Rio de Janeiro
Setembro de 2010

UMA COMPARAÇÃO DE MODELOS DE BRDFS PARA PHOTON MAPPING

Alberto Barbosa Júnior

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

Prof. Antonio Alberto Fernandes de Oliveira, D.Sc.

Prof. Ricardo Guerra Marroquim, D.Sc.

Prof. Claudio Esperança, Ph.D.

Prof. Anselmo Antunes Montenegro, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
SETEMBRO DE 2010

Barbosa Júnior, Alberto

Uma Comparação de Modelos de BRDFs para Photon Mapping/Alberto Barbosa Júnior. – Rio de Janeiro: UFRJ/COPPE, 2010.

IX, 61 p.: il.; 29, 7cm.

Orientadores: Antonio Alberto Fernandes de Oliveira
Ricardo Guerra Marroquim

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2010.

Referências Bibliográficas: p. 58 – 61.

1. Computação Gráfica. 2. Iluminação Global. I. Oliveira, Antonio Alberto Fernandes de *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

UMA COMPARAÇÃO DE MODELOS DE BRDFS PARA PHOTON MAPPING

Alberto Barbosa Júnior

Setembro/2010

Orientadores: Antonio Alberto Fernandes de Oliveira
Ricardo Guerra Marroquim

Programa: Engenharia de Sistemas e Computação

O método de *photon mapping*, ou mapeamento de fótons, é uma extensão de *Monte Carlo raytracing*, que provê eficiência enquanto mantém muitas vantagens das técnicas originais. Uma dessas vantagens é a possibilidade de simular muitos efeitos de iluminação global, pois qualquer tipo de BRDF pode ser utilizada. O objetivo desta dissertação é analisar e comparar resultados gerados por diferentes BRDFs no contexto de *photon mapping*.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

A COMPARISON OF BRDFS MODELS FOR PHOTO MAPPING

Alberto Barbosa Júnior

September/2010

Advisors: Antonio Alberto Fernandes de Oliveira
Ricardo Guerra Marroquim

Department: Systems Engineering and Computer Science

The photon mapping method is an extension of Monte Carlo raytracing techniques, providing computational eficiency while maintaining the advantages of original techniques. One of these advantages is the possibility to simulate all the efects of global illumination, since any type of BRDF can be handled. The objective of this work is to analyze and compare the results generated by different types of BRDFs in the context of photon mapping.

Sumário

Lista de Figuras

Capítulo 1

Introdução

A renderização realística, dado seus desafios e sua demanda em mercados como o cinematográfico e o de jogos, recebeu foco e um grande esforço nos últimos tempos. De forma sucinta, consiste na geração de imagens a partir de um conjunto de dados que compõem a descrição tridimensional de uma cena. Basicamente, é minimamente necessário tamanhos e posições dos objetos presentes e propriedades dos materiais que os constituem, posições e características das fontes de luz, e uma câmera virtual que determina o ponto de onde a cena está sendo observada. O objetivo é gerar imagens realísticas que não possam ser diferenciadas de fotografias.

Esta é uma tarefa difícil e desafiadora. Entre os estudos e áreas do conhecimento relacionadas, está a compreensão do comportamento da luz e sua interação com o ambiente, que é um dos focos deste trabalho. A partir dessa compreensão, é possível desenvolver modelos que descrevam este comportamento e aplicá-los a simulações. Na renderização realística a sub-área responsável por esses estudos e suas aplicações é definida como iluminação global.

1.1 Iluminação Global

Na computação gráfica, o objetivo das técnicas de iluminação global é computar não somente o efeito que as fontes de luz tem diretamente nos objetos presentes no ambiente (iluminação direta), mas também a forma com que as reflexões dessa luz na superfície de cada um dos objetos afeta os demais (iluminação indireta).

Nas últimas décadas, várias técnicas foram desenvolvidas com esse propósito, sendo a grande maioria delas derivações de dois métodos precursores: *raytracing* e radiosidade. Posteriormente, foram desenvolvidos métodos híbridos que mesclam estas duas técnicas, buscando aproveitar as suas vantagens de forma complementar.

1.1.1 Raytracing

O algoritmo de *raytracing* foi apresentado em 1980 por Turner Whitted [2] e consiste em traçar raios partindo de um observador através da cena em que ele se encontra até as fontes de luz presentes.

O algoritmo, que se tornou um dos mais populares no campo da computação gráfica, foi estendido ao longo dos anos. Em sua versão clássica não soluciona completamente o problema da iluminação global. Os raios traçados pela cena determinam a visibilidade em um dado pixel, e computam reflexões e refrações especulares além da iluminação direta.

Para computar a iluminação global simulando efeitos como iluminação indireta, cáusticas, *motion blur* e reflexões *glossy* o algoritmo foi estendido utilizando métodos de Monte Carlo (e.g. [3]), e consequentemente foram nomeados como Monte Carlo *raytracing*. A ideia central é distribuir os raios estocasticamente pela cena para computar de forma aproximada todos os caminhos percorridos pela luz. Esses métodos são discutidos de forma mais ampla no capítulo 3.

1.1.2 Radiosidade

Radiosidade é um método de elementos finitos, apresentado originalmente por Gorat et al. [4], como uma alternativa ao algoritmo clássico de *raytracing*, dado que neste, efeitos da iluminação global provenientes de reflexões difusas são difíceis de serem computados.

Primeiramente, as superfícies dos objetos presentes na cena são divididas em elementos denominados *patches*, ou retalhos. A luz proveniente de cada um desses *patches* é computada, tornando possível o cálculo da distribuição da iluminação no ambiente. Diferentemente do algoritmo de *raytracing* que possui complexidade relacionada ao espaço de imagem, o algoritmo de radiosidade atua diretamente no espaço do objeto. Isto permite o cálculo da solução de forma independente da posição do observador, sendo vantajoso para aplicações onde existe a necessidade de navegação pelo ambiente. Por outro lado, a solução está diretamente ligada a complexidade da cena a ser tratada podendo exigir um grande esforço de pré-processamento.

Inicialmente o método foi utilizado em cenas que continham somente superfícies difusas. Ao longo do tempo, a técnica foi estendida para utilização de modelos mais complexos de iluminação. Porém, os métodos baseados em radiosidade tem a limitação de não tratar bem a iluminação em superfícies especulares.

1.1.3 Métodos Híbridos

Como dito, os algoritmos de *raytracing* lidam bem com reflexões especulares, enquanto os de radiosidade com reflexões difusas. Um caminho natural a ser seguido é a utilização de ambos no cálculo da distribuição da iluminação para se aproveitar o melhor dos dois mundos, os chamados métodos híbridos [5] [6] [7].

Existem diversas variações, porém, de forma geral, todos seguem uma estratégia de dois passos. Em um primeiro passo, a iluminação indireta difusa é calculada utilizando um algoritmo de radiosidade, seguido do cálculo da luz transportada nas reflexões e refrações especulares, utilizando *raytracing*, levando em consideração os valores calculados no passo anterior.

Photon Mapping

O algoritmo de *photon mapping*, ou mapeamento de fótons, pode ser considerado uma variação dos métodos híbridos. Porém, o cálculo de parte da iluminação no primeiro passo não é feito utilizando radiosidade. Fótons são lançados pela cena a partir dos pontos de luz e suas interações são armazenadas em uma estrutura de dados independente da geometria: o *photon map*. Essa informação é utilizada posteriormente no cálculo da iluminação global da cena.

O algoritmo, que é o utilizado neste trabalho, é uma extensão do Monte Carlo *raytracing*, mantendo as vantagens advindas do mesmo, porém, buscando uma maior eficiência computacional. O mesmo está descrito de forma mais detalhada no capítulo 4.

1.1.4 Modelos de Iluminação

Como dito anteriormente, para se calcular corretamente a iluminação global em uma cena, é necessário que o comportamento da luz e sua interação com o ambiente sejam compreendidos. Modelos que descrevem esse comportamento foram desenvolvidos ao longo dos anos (e.g. [8] [9] [10]) e definem, basicamente, o quanto e em que direção a luz incidente em uma superfície é refletida de volta ao ambiente.

Nesse trabalho, como já mencionado, utilizamos o algoritmo de *photon mapping* e são os modelos de iluminação que proveem o formalismo necessário para a simulação da interação dos fótons com o ambiente e para o cálculo da radiância refletida nas superfícies.

No capítulo 2, são introduzidos conceitos relacionados aos modelos de iluminação, enquanto no capítulo 5 os modelos utilizados no trabalho são descritos.

1.2 Proposta do Trabalho

Este trabalho tem como objetivo principal o estudo de alguns modelos de iluminação aplicados ao escopo do *photon mapping*, fazendo uma análise comparativa dos resultados alcançados pela utilização de cada um deles.

Etapas intermediárias a esse objetivo também são discutidas, como a implementação dos algoritmos envolvidos e suas estruturas de dados, objetivando um entendimento prático de toda teoria apresentada.

1.3 Organização da Tese

Além dessa introdução, o trabalho está organizado da seguinte maneira:

- Capítulo 2: é apresentada uma pequena introdução à física envolvida na iluminação global.
- Capítulo 3: neste capítulo os métodos de *Monte Carlo Raytracing*, precursores ao *photon mapping*, são introduzidos.
- Capítulo 4: aqui o algoritmo de *photon mapping* é detalhado assim como a estrutura de dados utilizada no mesmo, o *photon map*.
- Capítulo 5: aqui os modelos de iluminação utilizados neste trabalho são descritos.
- Capítulo 6: o foco deste capítulo são os detalhes da implementação tanto do método de *photon mapping* quanto dos modelos de iluminação detalhados nos capítulos anteriores.
- Capítulo 7: os resultados da implementação do trabalho são mostrados e analisados neste capítulo.
- Capítulo 8: neste capítulo são apresentadas as conclusões e sugestões de trabalhos futuros.

Capítulo 2

Fundamentos da Iluminação Global

Como dito anteriormente, o passo primordial para a simulação da iluminação global, é o entendimento físico do comportamento da luz para que modelos que o representam possam ser desenvolvidos. Neste capítulo são apresentados alguns conceitos fundamentais relacionados a luz.

2.1 Radiometria

A área que envolve as quantidades físicas que representam a energia da luz e suas medições é a radiometria, e alguns conceitos básicos são apresentados a seguir.

2.1.1 Conceitos Básicos de Radiometria

Energia Radiante ou Fluxo Quantidade radiométrica fundamental, expressa o fluxo total de energia passando por uma superfície por unidade de tempo. É denotado por Φ e expresso em *Watt(W)*.

Irradiância É a energia radiante em uma superfície por unidade de área. A irradiância E é expressa em $Watt/m^2$, e dada por:

$$E = \frac{d\Phi}{dA}. \tag{2.1}$$

Intensidade de Radiação Denotado por I, é o fluxo radiante por unidade de ângulo sólido $d\vec{\omega}$,

$$I(\vec{\omega}) = \frac{d\Phi}{d\vec{\omega}}. \tag{2.2}$$

Radiância Expressa quanta energia chega, ou deixa um determinado ponto em uma superfície, por unidade de ângulo sólido, por unidade de área projetada, e é medida em $W/(esteroradianos \cdot m^2)$. É considerada a quantidade mais importante na iluminação global, pois captura a aparência de um objeto. A radiância L é dada por

$$L(x, \vec{\omega}) = \frac{d^2\Phi}{\cos\theta dA d\vec{\omega}}. \tag{2.3}$$

2.2 Interação entre a Luz e as Superfícies

A luz emitida em uma cena interage com o ambiente, e a cada interação pode ser refletida, transmitida ou absorvida nas superfícies do objeto com que está interagindo. A maneira como essa interação é formalizada é apresentada na subseção 2.2.1.

2.2.1 BSDF

No caso mais geral, a luz incide em uma superfície em um ponto $x^{'}$, em uma direção $\vec{\omega}^{'}$, e pode ser refletida de volta ao ambiente em um ponto x em uma direção $\vec{\omega}$, devido a possibilidade de reflexão abaixo da superfície. A função que define essa relação entre radiância incidente e refletida é chamada BSSRDF (*bidirectional scattering surface reflectance distribution function*). Em grande parte dos trabalhos em computação gráfica (inclusive neste), assume-se que a luz é refletida no mesmo ponto da superfície em que incidiu, permitindo uma simplificação dos cálculos, resultando em ganho computacional. A descrição das propriedade de reflectância em uma superfície, considerando esta simplificação, é dada por uma função denominada BRDF(*bidirectional reflectance distribution function*).

Uma BRDF provê o formalismo necessário para descrever as reflexões da luz em superfícies, simulando propriedades específicas de materiais que possam constituir os objetos. Uma BRDF $f_r(x, \vec{\omega}^{'}, \vec{\omega})$, determina o quanto da luz incidente no ponto x e na direção $\vec{\omega}^{'}$, é irradiada na direção $\vec{\omega}$, como ilustrado na Figura 2.1. Uma BRDF em um ponto x é determinada como a relação entre a radiância refletida e a irradiância:

$$\begin{aligned} f_r(x, \vec{\omega}, \vec{\omega}^{'}) &= \frac{dL_r(x, \vec{\omega})}{dE_i(x, \vec{\omega}^{'})} \\ &= \frac{dL_r(x, \vec{\omega})}{L_i(x, \vec{\omega}^{'})(\vec{\omega^{'}} \cdot \vec{n})d\omega^{'}} \end{aligned} \tag{2.4}$$

onde $\vec{n}$ é a normal em x.

Para se calcular a radiância refletida $L_r(x, \vec{\omega})$ em um ponto x, pode-se integrar

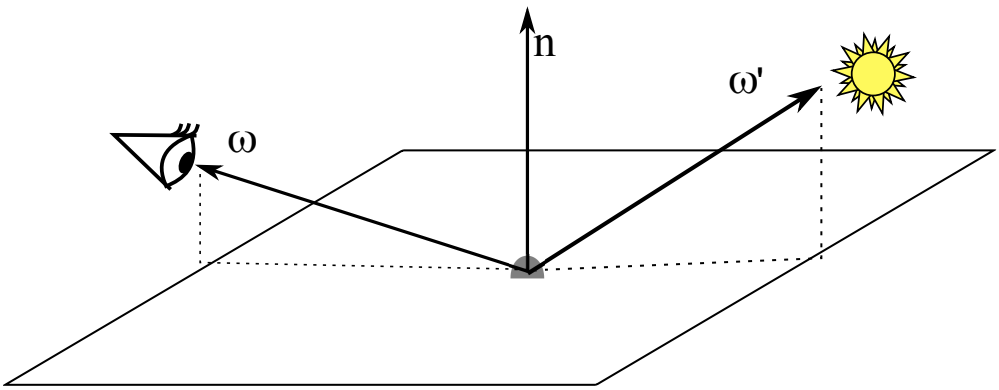


Figura 2.1: As direções de incidência e irradiância de uma BRDF.

a BRDF sob o hemisfério de direções da radiância incidente Ω,

$$\begin{aligned} L_r(x,\vec{\omega}) &= \int_\Omega f_r(x,\vec{\omega}',\vec{\omega})dE(x,\vec{\omega}') \\ &= \int_\Omega f_r(x,\vec{\omega}',\vec{\omega})L_i(x,\vec{\omega}')(\vec{\omega}'\cdot\vec{n})d\vec{\omega}'. \end{aligned} \tag{2.5}$$

Uma BRDF tem algumas importantes propriedades. O seu valor continua igual, mesmo trocando as direções de chegada e saída. Essa propriedade é chamada reciprocidade de *Helmhotz*, e significa que revertendo a direção da luz, a quantidade refletida continua a mesma:

$$f_r(x,\vec{\omega}',\vec{\omega}) = f_r(x,\vec{\omega},\vec{\omega}'). \tag{2.6}$$

Outra propriedade importante e a conservação da energia, que determina que a quantidade de energia refletida é sempre menor ou igual à energia incidida. Determina também que toda BRDF precisa satisfazer

$$\int_\Omega f_r(x,\vec{\omega}',\vec{\omega})(\vec{\omega}'\cdot\vec{n})d\omega' \leq 1, \forall\ \vec{\omega'}. \tag{2.7}$$

A luz além de refletir em uma superfície, pode também ser transmitida. Esse tipo de interação, onde geralmente $\vec{\omega}'$ e $\vec{\omega}$ estão em hemisférios opostos, é descrita por uma BTDF(*bidirectional transmittance distribution function*). Esta função pode ser definida de maneira semelhante a uma BRDF, não obedecendo porém a regra de reciprocidade. Por conveniência, muitos trabalhos agrupam as BRDFs e BTDFs em um único grupo denominado BSDF (*bidirectional scattering distribution function*).

2.3 Equação de Renderização

Originalmente apresentada por Kajiya [11] a equação de renderização é a fundamentação matemática para todos os algoritmos de iluminação global, e expressa

o equilíbrio da distribuição da energia da luz em uma cena. Assumindo-se a inexistência da influência do meio (*participating media*), pode ser usada para computar a radiância emanada em qualquer local das superfícies de um modelo. A radiância emanada L_o é a soma da radiância emitida L_e e a radiância refletida L_r,

$$L_o(x, \vec{\omega}) = L_e(x, \vec{\omega}) + L_r(x, \vec{\omega}). \tag{2.8}$$

Com o uso da equação 2.5 para calcular a radiância refletida tem-se,

$$L_o(x, \vec{\omega}) = L_e(x, \vec{\omega}) + \int_{\Omega} f_r(x, \vec{\omega}^{'}, \vec{\omega}) L_i(x, \vec{\omega}^{'})(\vec{\omega}^{'} \cdot \vec{n}) d\vec{\omega}^{'}. \tag{2.9}$$

Esta é a forma como a equação da renderização é utilizada nos métodos baseados em *Monte Carlo raytracing* de uma forma aproximada, como detalhado no capítulo 3. Note que esta integral possui um caráter recursivo, já que para avaliar L_i é preciso avaliar a integral em um determinado ponto $x^{'}$, de onde o raio foi irradiado na direção $-\vec{\omega}^{'}$.

Capítulo 3

Monte Carlo Raytracing

A técnica utilizada neste trabalho, o *photon mapping* é uma extensão das técnicas de *Monte Carlo raytracing* [12]. Por sua vez, os métodos de *Monte Carlo raytracing* são extensões do algoritmo clássico de *raytracing*, permitindo simular efeitos de iluminação global, como *color bleeding*, *soft shadows,* iluminação indireta, entre outros. Enquanto no algoritmo clássico, a informação da radiância é calculada ao longo de apenas uma amostra (um raio), o que na prática não é uma boa aproximação, nos algoritmos de *Monte Carlo raytracing* um número maior de amostras é utilizado. Métodos estocásticos são incluídos no cálculo da interação da luz com o ambiente, permitindo a simulação aproximada de todos os caminhos percorridos pela luz em uma determinada cena.

Como relatado por Jensen [13], existem várias vantagens em relação a outros métodos, como os baseados em elementos finitos, tais como:

- qualquer tipo de geometria pode ser tratada;
- qualquer tipo de BRDF pode ser tratada;
- reflexões especulares são fáceis de serem calculadas em qualquer superfície;
- o consumo de memória é baixo;
- a precisão é controlada no nível do pixel/imagem.

O problema é que a integral de iluminação (equação 2.9) não pode ser resolvida de forma exata para distribuições arbitrárias. Adicionalmente, seu caráter recursivo acentua ainda mais a sua inviabilidade computacional. A integração de Monte Carlo permite que esta integral seja discretizada, sendo aproximada então por um somatório finito.

Em resumo, a integração de Monte Carlo aproxima uma integral definida do tipo:

$$M = \int_a^b f(x)dx, \tag{3.1}$$

por uma estimativa avaliada em n amostras randômicas uniformemente distribuídas:

$$< M >= \frac{b-a}{n} \sum_{i=1}^{n} f(x_i). \tag{3.2}$$

O número de amostras avaliadas durante este somatório está diretamente relacionado com a qualidade da aproximação da integral, ou seja, ao tender a infinito a solução se aproxima da solução exata. A Figura 3.1 ilustra a amostragem de uma função.

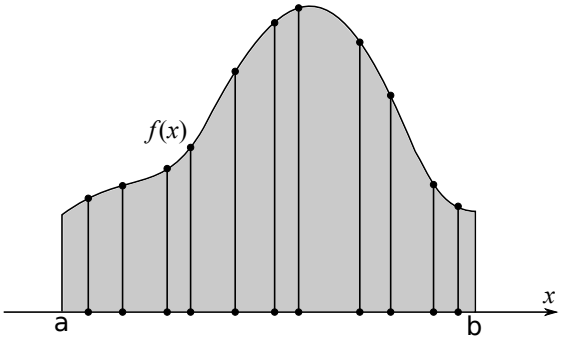


Figura 3.1: Amostragem uniformemente distribuída de uma função.

O principal problema dos algoritmos de *Monte Carlo raytracing* é a variância representada por ruídos nas imagens renderizadas. A convergência dos métodos de *Monte Carlo* é lenta, tornando necessário um grande número de amostras para eliminar esses ruídos, o que torna o algoritmo custoso computacionalmente. De uma forma geral, o desvio padrão possui uma relação quadrática em relação ao número de amostras, i.e., para reduzir o erro pela metade é necessário usar quatro vezes mais amostras.

Outras formas mais eficazes de reduzir os ruídos são propostas na literatura, como por exemplo formas de amostragem mais eficientes. Entre estas, uma das mais conhecidas e utilizada é a de amostragem por importância [14, 15]. A ideia destes métodos é favorecer a escolha de amostras que contribuam mais com a redução da variância, ou seja, escolher amostras onde a área da função é grande, como ilustrado na Figura 3.2.

Entre os métodos classificados como *Monte Carlo raytracing* estão o *raytracing* distribuído [3], *path tracing* [11], *path tracing bidirecional* [16] e o método *Metropolis* de transporte da luz [17]. Para ilustrar esse conjunto de técnicas, o algoritmo de

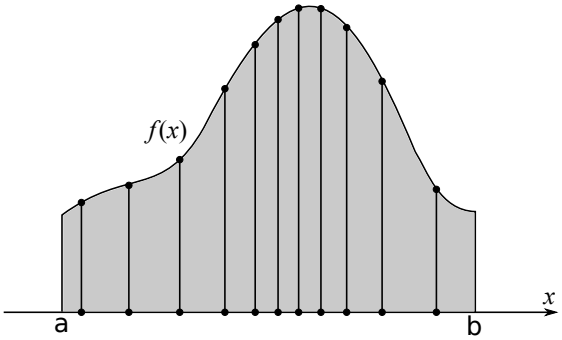


Figura 3.2: Ilustração da mesma função da Figura3.1 com uma amostragem por importância.

path tracing será apresentado na seção 3.1.

3.1 Path Tracing

Apresentado no mesmo trabalho [11] onde foi apresentada a equação de renderização, como uma solução para mesma, o algoritmo de *path tracing* é uma extensão simples do algoritmo clássico de *raytracing*, mas que permite o cálculo de todos os efeitos da iluminação global por amostrar estocasticamente todos os caminhos da luz.

Seguindo a ideia comum dos métodos de *Monte Carlo raytracing*, no algoritmo de *path tracing* a multi-amostragem ocorre no espaço de pixel. Um número pré-determinado de raios com origem no observador são lançados com direção aleatória em cada pixel da imagem resultante, interagindo com o ambiente de forma recursiva até alcançar uma fonte de luz presente na cena. O valor da radiância referente a cada raio é acumulado durante as interações do mesmo com o ambiente. A cor de cada pixel é determinada pela média dos resultados encontrados por cada raio referente a ele.

Na figura 3.3 é apresentado o pseudo-código para o algoritmo *path tracing*.

```
RenderizarImagem
    Foreach pixel Do
        i ⇐ 0;
        cor ⇐ 0;
        While i < número de amostras Do
            gerar uma amostra para o pixel;
            lançar um raio do observador ao ponto da amostra no pixel;
            cor ⇐ cor + Trace(raio);
            i++;
        cor-pixel ⇐ cor/#amostras;
End RenderizarImagem

Trace(raio)
    encontrar a interseção mais próxima com a cena;
    calcular ponto de interseção e normal;
    cor ⇐ Shade(ponto, normal);
    Return cor;
End Trace

Shade(ponto, normal)
    cor ⇐ 0;
    Foreach fonte de luz Do
        testar visibilidade entre fonte de luz e ponto;
        If (visível)
            cor ⇐ cor + iluminação direta;
    cor ⇐ Trace(raio randômico refletido);
End Shade
```

Figura 3.3: *Path Tracing*

Capítulo 4

Photon Mapping

O algoritmo de *photon mapping* foi introduzido por Jensen [18] [19] e teve como motivação a busca por uma simulação eficiente de todos os efeitos da iluminação global, independente da complexidade do modelo tratado. Na prática, isso significa que o algoritmo deveria lidar com qualquer tipo de geometria e qualquer tipo de BRDF (*bidirectional reflectance distribution function*, ver detalhes no capítulo 5).

Dado o objetivo proposto e fazendo uma revisão nos algoritmos até então desenvolvidos, observa-se que os únicos métodos com tais características são os baseados em Monte Carlo *raytracing* [13]. Porém, o principal problema desses métodos é o ruído que as imagens resultantes contêm devido a aproximação da integral, como relatado no capítulo 3. A eliminação desse ruído é extremamente custoso, dado que, para isso, é necessário aumentar substancialmente o número de raios lançados.

O mérito do algoritmo de *photon mapping* é estender os métodos puramente baseados em Monte Carlo *raytracing*, mantendo as suas vantagens e ao mesmo tempo provendo eficiência e possibilitando resultados mais precisos. Como já dito anteriormente, o algoritmo de *photon mapping* consiste de dois passos: o passo de *photon tracing* e o de renderização. Estes passos e a estrutura de dados utilizada para armazenar informações sobre a iluminação serão explicados a seguir.

4.1 Photon Tracing

Photon tracing, ou traçado de fótons, é o processo que consiste na emissão de fótons a partir das fontes de luz e do computo das suas interações com o ambiente. As informações advindas desse processo são armazenadas na estrutura de dados *photon map* (ver seção 4.2), e posteriormente utilizadas no passo de renderização para o cálculo da radiância refletida nas superfícies.

O funcionamento do *photon tracing* é muito similar ao do *raytracing*, diferindo no fato de que os fótons propagam o fluxo da luz, enquanto os raios recuperam a informação de radiância.

4.1.1 Emissão de Fótons

Para cada fonte de luz presente na cena é gerado um número de fótons, e a energia total dessa fonte de luz é distribuída entre eles. Este número geralmente varia de milhares a dezenas de milhares, e está diretamente relacionado à qualidade da radiância estimada no passo de renderização. É necessário uma densidade de fótons nas superfícies suficientemente grande para alcançar um resultado suave, ou seja, livre de artefatos de interpolação.

Existem diversos tipos de luz, e os fótons são gerados de formas diferentes para cada uma delas. Alguns tipos são: pontuais, direcionais, esféricas e quadrangulares, entre outras. A seguir, será explicado como esta geração ocorre nos dois tipos de luz implementados nesse trabalho: a luz pontual e a luz quadrangular.

Luz Pontual

A luz pontual é a mais simples de todas: os fótons são emitidos em todas as direções uniformemente. Entre as diversas formas de se fazer a amostragem dos fótons a partir de uma luz pontual, está a amostragem por rejeição (ou *rejection sampling*), que foi utilizada neste trabalho. Pontos são gerados randomicamente em um cubo unitário, e se o ponto estiver contido na esfera unitária o mesmo é utilizado como a direção de lançamento do fóton, caso contrário, é rejeitado e um novo ponto é gerado. O algoritmo 4.1 descreve esse processo.

$n_e \Leftarrow 0$; // *número de fótons emitidos*
While (não foram lançados fótons suficientes) **Do**
  Repeat
    $x \Leftarrow 2\xi_1 - 1$;//$\xi_1 \in [0,1]$ *é um número randômico*
    $y \Leftarrow 2\xi_2 - 1$;//$\xi_2 \in [0,1]$ *é um número randômico*
    $z \Leftarrow 2\xi_3 - 1$;//$\xi_3 \in [0,1]$ *é um número randômico*
  Until $(x^2 + y^2 + z^2 > 1)$
  $\vec{d} \Leftarrow < x, y, z >$;
  $\vec{p} \Leftarrow$ posição da fonte de luz;
  traçar fóton de $\vec{p}$ em direção a $\vec{d}$;
  $n_e \Leftarrow n_e + 1$;
end While
dividir a energia dos fótons gerados por $1/n_e$;

Figura 4.1: Geração de fótons a partir de fonte de luz pontual.

Luz Quadrangular

A luz quadrangular, como o próprio nome sugere, consiste em uma superfície retangular, de onde é possível emitir fótons de qualquer ponto e em qualquer direção,

contanto que esta esteja no mesmo hemisfério de sua normal. A geração dos fótons é feita de forma bastante intuitiva. Uma posição na superfície é escolhida de forma aleatória, e partindo daquele ponto é, então, escolhida uma direção seguindo o modelo anterior para luz pontual, porém restrito ao hemisfério superior.

Os tipos de luz que cobrem uma área superficial possuem a propriedade de gerar sombras suaves, isto é, regiões onde a luz está parcialmente visível. Estas regiões são chamadas de penumbra, ao contrário das regiões que se encontram completamente bloqueadas por algum objeto na cena, como ilustrado na Figura 4.2.

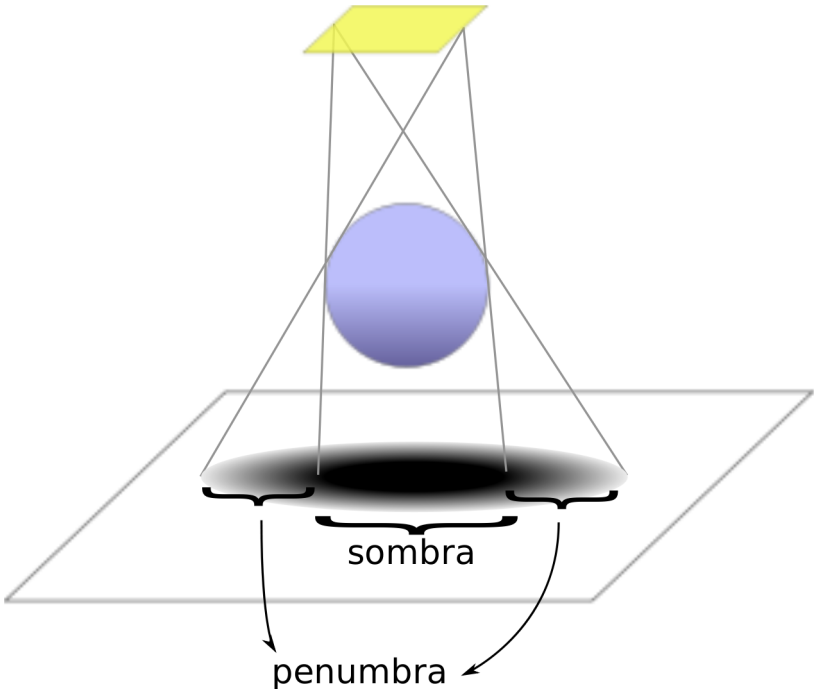


Figura 4.2: A esfera gera uma região de sombra e de penumbra na superfície devido à emissão de luz a partir de uma área quadrangular.

4.1.2 Espalhamento de Fótons

A partir do momento em que um fóton é emitido por uma fonte de luz, o mesmo pode intersectar uma superfície ou sair do ambiente em questão. De acordo com as propriedades do material, em cada interseção parte da energia do fóton pode ser refletida, transmitida ou absorvida. Uma forma de simular esse fenômeno seria a criação de fótons com a energia referente a cada um desses tipos de interação, porém, isso seria custoso computacionalmente e em relação a memória utilizada para armazenamento dessa informação. Para contornar esse problema, é utilizado uma técnica chamada *roleta russa*, onde o fóton mantem o mesmo valor de energia, e probabilisticamente é decidido se ele será refletido, transmitido ou absorvido.

Roleta Russa

Introduzida na computação gráfica por Arvo e Kirk [20], é uma técnica estocástica inicialmente desenvolvida para prover eficiência na computação da física de partículas. Pode ser definida como uma técnica de amostragem onde uma função de distribuição de probabilidades é usada para eliminar partes não importantes do domínio.

No *photon tracing*, esta técnica permite a exclusão de fótons com pouca relevância e consequentemente ganho de eficiência computacional. Além disso, possibilita que os fótons armazenados no *photon map* tenham a mesma energia, o que permite boa qualidade na estimativa da radiância utilizada no passo de renderização. A seguir, dois usos da técnica são exemplificados.

Reflexão ou Absorção? Dado um fóton com energia Φ_p incidente em uma superfície com material com constante de reflectividade d, o método da roleta russa pode ser utilizado para definir se tal fóton será absorvido ou refletido. O algoritmo 4.3 mostra como tal decisão é tomada.

```
p ⇐ d; // probabilidade de reflexão
ξ ⇐ random(); //ξ ∈ [0,1] é um número randômico distribuído uniformemente
If(ξ < d)
  fóton é refletido com energia Φ_p;
Else
  fóton é absorvido
End If
```

Figura 4.3: Reflexão ou Absorção?

A técnica, apesar de aparentemente simples, permite que o resultado correto seja calculado de uma forma mais eficiente, como pode ser verificado no seguinte exemplo. Dada uma superfície que tenha um material com reflectividade 0.5, 1000 fótons são lançados sobre ela a partir de uma fonte de luz. Ao invés dos 1000 fótons serem refletidos com metade da energia incidente, a roleta russa permite que probabilisticamente, 500 deles sejam refletidos sem perda de energia.

Reflexão Difusa ou Especular? A técnica de roleta russa também pode ser utilizada para decidir o tipo de reflexão de um fóton em uma superfície, quando esta conta com mais de um tipo de reflexão. Por exemplo, em uma superfície com reflexões difusa e especular, o tipo de reflexão a ser usado no espalhamento do fóton é determinado segundo o algoritmo 4.4. Também neste exemplo o fóton refletido não tem sua energia modificada. O resultado correto é obtido devido a média das interações dos fótons ao longo do tempo.

```
ξ ⇐ random(); // ξ ∈ [0, 1] é um número randômico distribuído uniformemente
If(ξ ∈ [0, ρd])
  fóton é refletido de forma difusa;
Else If (ξ ∈ [ρd, ρs + ρd])
  fóton é refletido de forma especular
Else If (ξ ∈ [ρs + ρd, 1])
  fóton é absorvido
End If
```

Figura 4.4: Reflexão Difusa ou Especular?

Modelos de Iluminação

Para se definir como um fóton é distribuído, além dos meios para determinar se o mesmo será refletido, transmitido ou absorvido ao atingir uma superfície, é preciso determinar como será realizada essa interação. Cada superfície, dado as propriedades do seu material segue um modelo pré-determinado de interação com a luz. Esses modelos de iluminação serão discutidos amplamente no capítulo 5.

4.1.3 Armazenamento dos Fótons

O conjunto dos fótons lançados a partir das fontes de luz, e suas interações com o ambiente, representam uma aproximação da iluminação indireta. A medida que esta informação é gerada no *photon tracing*, é necessário armazená-la em uma estrutura de dados para ser utilizada posteriormente no passo de renderização, onde será integrada ao cálculo da iluminação global.

A interação de um fóton com uma superfície só é armazenada nessa estrutura de dados quando esta superfície for difusa. Fótons não são armazenados em superfícies especulares, pois a chance de um fóton intersectar um ponto dessa superfície vindo de uma direção especular é muito pequena. A melhor forma de renderizar reflexões especulares é utilizando o algoritmo de *raytracing*. Nesse trabalho, informações sobre a luz direta, ou seja, a que incide diretamente das fontes de luz nas superfícies, é calculada da mesma forma que no algoritmo clássico de *raytracing*. Portanto, a primeira interação de cada fóton, que representa essa informação, também não é armazenada na estrutura. Mais detalhes serão explicados na seção 4.3.

A estrutura de dados onde os fótons são armazenados é chamada *photon map*, ou mapa de fótons.

4.2 Photon Map

Finalizado o passo de *photon tracing*, tem-se armazenado na estrutura de dados as informações referentes aos fótons lançados. Pode-se inferir algumas características desejadas da estrutura de dados a ser utilizada analisando os seguintes fatos: em um caso geral, o número de fótons lançados é alto; a distribuição espacial dos fótons não é uniforme; e a informação sobre a iluminação em um determinado ponto é recuperada a partir de uma busca dos fótons vizinhos. Desta forma, a estrutura deve ser compacta, lidar bem com dados espaciais não uniformes e permitir uma busca eficiente de vizinhos mais próximos. A estrutura proposta por Jensen, e também utilizada neste trabalho, é a *kd-tree*.

A *kd-tree* é uma árvore binária, onde cada nó é um ponto k-dimensional. Adaptada ao contexto do *photon map*, os nós são representados pelos fótons, em um espaço tridimensional. Cada nó interno da *kd-tree* divide o espaço com um plano ortogonal a um dos eixos coordenados. Todos os pontos abaixo desse plano fazem parte da sub-árvore esquerda desse nó, e os pontos acima desse plano da sub-árvore direita.

No *photon map* a principal busca são pelos k fótons vizinhos mais próximos. Em uma *kd-tree* não balanceada a busca pelo fóton mais próximo de um determinado ponto tem a complexidade $O(n)$ no pior caso. Porém em uma *kd-tree* balanceada essa complexidade em pior caso é $O(n^{1-1/d})$, sendo d a dimensão da *kd-tree*. Na busca dos pelos k fótons vizinhos mais próximos, a complexidade então é $O(k + n^{1-1/d})$ no pior caso, porém como demonstrado por Bentley [21] se mantém em média com complexidade $O(k + log\ n)$.

A implementação da estrutura de dados utilizada nesse trabalho é baseada na implementação apresentada por Jensen [13]. Detalhes referentes a essa implementação são apresentados a seguir.

4.2.1 Representação do Fóton

Com a necessidade de armazenar milhões de fótons no *photon map*, é necessário que as informações referentes ao mesmo sejam representadas de forma a ocupar pouco espaço em memória. Jensen [13] propõe a estrutura mostrada na figura 4.5 que utiliza 20 bytes para representar um fóton.

Na estrutura, a energia do fóton é representada de forma compacta com 4 *bytes* como proposto por Ward [22], que pode ser substituído pela representação tradicional com 3 *floats*, caso a memória não seja uma limitação. A direção incidente, utilizada para computar a contribuição dos fótons em superfícies não difusas e verificar em qual lado de uma superfície o fóton incidiu, é representado em coordenadas esféricas utilizando 2 *bytes*. Uma tabela de conversão é utilizado para evitar que

```
Structure fóton
  float x,y,z; // posição
  char p[4]; // energia
  char phi, theta; // direção de incidência
  short flag; // flag usado na kd-tree
End Structure
```

Figura 4.5: Estrutura de Dados de um Fóton

essa conversão se repita diversas vezes durante o tempo de execução, por se tratar de uma operação custosa envolvendo o uso de funções como `cos()` e `sin()`.

4.2.2 Kd-Tree Balanceada

No início da subseção sobre o *photon map*, foram discutidos os tempos em pior caso para se efetuar buscas em uma *kd-tree* balanceada e não balanceada. Por outro lado a complexidade do algoritmo de balanceamento é *O(n log n)*. Porém, como são realizadas diversas buscas contra apenas um balanceamento inicial, fica evidente a vantagem da *kd-tree* balanceada neste contexto.

Antes da discussão sobre o funcionamento do algoritmo de balanceamento, é importante saber que, assim como proposto por Jensen [13], a *kd-tree* foi implementada utilizando uma estrutura baseada em *heap*. Isso permite uma economia de memória pois a relação entre os nós é dado de forma implícita, não sendo necessária a utilização de ponteiros. A estrutura *heap* consiste em um array de dados, ordenados de forma a caracterizar a relação entre seus elementos. O primeiro elemento da lista é a raiz da árvore, e um elemento em uma posição i tem o primeiro elemento de sua sub-árvore esquerda localizado na posição $2i$ e o primeiro elemento da sub-árvore direita localizado na posição $2i + 1$. O fato da *kd-tree* estar balanceada garante que não existam posições vazias no array nessa representação de dados. As relações sendo implícitas, torna possível uma economia de 40% no uso de memória (considerando a estrutura de 20 bytes demonstrada na subseção 4.2.1), valor significativo dado que são armazenados quantidades expressivas de fótons.

O algoritmo 4.6 ilustra o balanceamento da *kd-tree*.

4.2.3 Busca no Photon Map

A busca no *photon map* é uma operação crítica no algoritmo, pois além de envolver um espaço com milhões de fótons, é repetida diversas vezes para o cálculo da radiância como mostrado na subseção 4.3.1. Devido a essas considerações, o algoritmo de busca utilizado no trabalho e proposto por Jensen [13] tem algumas especificidades objetivando eficiência.

```
Balancear(fótons[])
  encontrar o cubo que envolve todos os fótons;
  selecionar a maior dimensão dim do cubo;
  encontrar a mediana dos fótons em dim;
  s1 ⇐ todos os pontos abaixo da mediana;
  s2 ⇐ todos os pontos acima da mediana;
  no ⇐ mediana;
  no.esquerdo ⇐ Balancear(s1);
  no.direito ⇐ Balancear(s2);
  retornar no;
End Balancear
```

Figura 4.6: Balanceamento do *Photon Map*

A busca realizada são pelos n fótons mais próximos de um determinado ponto no espaço. Para efetuar essa busca é definido um raio máximo, o que especialmente no caso do *photon mapping* faz sentido por não permitir que fótons muito distantes do ponto avaliado contribuam para o cálculo da sua iluminação. Isso significa que nem sempre o resultado irá conter n fótons, porém evita que grande parte da *kd-tree* seja percorrida quando são avaliados pontos em regiões com baixa densidade de fótons.

A partir do momento em que o número máximo n de elementos no resultado parcial é atingido, e a cada vez que um elemento mais próximo é incluído no resultado, elimina-se o mais distante e redimensiona-se o raio de busca, atribuindo-lhe o valor da maior distância entre os elementos contidos no resultado. Isso permite que o raio de busca diminua gradativamente, tornando a operação mais eficiente.

Algoritmos clássicos de busca por vizinhos próximos utilizam uma lista ordenada pela distância ao ponto avaliado, para permitir que o vizinho mais distante seja eliminado no caso da lista de resultados já conter n elementos, e um elemento mais próximo ser encontrado. Ao invés de ordenar a lista durante a execução, a busca no *photon map* utiliza a estrutura *heap máximo* [23], que provê a informação do vizinho mais distante de forma mais eficiente.

Uma última consideração simples, porém importante, é a utilização das distâncias ao quadrado, eliminando o cálculo da raiz quadrada, por não haver necessidade das distâncias reais durante a operação de busca.

Dado um *photon map*, uma posição x e uma distância máxima de busca d^2, o algoritmo 4.7 retorna um *heap* h com os fótons mais próximos. A primeira chamada ao algoritmo é *LocalizarFotons(1)* para inicializar a busca no nó raiz da *kd-tree*.

```
LocalizarFotons(p)
    If (2p + 1 < número de fótons)
        δ ⇐ distância ao plano de particionamento do nó n;
        If (δ < 0)
            // Lado esquerdo do plano, buscar na subárvore esquerda primeiro
            LocalizarFotons(2p);
            If (δ² < d²)
                // Buscar na subárvore direita
                LocalizarFotons(2p + 1);
        Else
            // Lado direito do plano, buscar na subárvore direita primeiro
            LocalizarFotons(2p + 1);
            If (δ² < d²)
                // Buscar na subárvore esquerda
                LocalizarFotons(2p);
    δ² ⇐ distância ao quadrado do fóton p a x
    // O fóton está próximo o suficiente?
    If (δ² < d²)
        inserir fóton no heap máximo h
        // Ajustar a distância máxima da busca
        d² ⇐ distância ao quadrado ao fóton na raiz do nó em h
End LocalizarFotons
```

Figura 4.7: Localização dos Fótons mais próximos

4.3 Renderização

Após fótons serem traçados e armazenados no *photon map*, é possível renderizar a cena utilizando as informações até então obtidas. Nesta etapa são utilizados de forma conjunta o algoritmo de *raytracing* e o *photon map*. Esse processo é explicado com mais detalhes posteriormente, entretanto é necessária uma discussão em torno do processo para calcular a radiância refletida nas superfícies.

4.3.1 Cálculo da Radiância com Photon Map

No passo de renderização é usada uma aproximação da equação de renderização para calcular a radiância refletida nas superfícies. Como explicado no capítulo 2, a equação de renderização é tipicamente descrita como:

$$L_r(x, \vec{\omega}) = \int_{\Omega_\infty} f_r(x, \vec{\omega}, \vec{\omega}') L_i(x, \vec{\omega}')(\vec{n}_x \cdot \vec{\omega}') d\vec{\omega}' \tag{4.1}$$

onde:
x é a localização na superfície,
$\vec{\omega}'$ é a direção de entrada da radiância,
$\vec{\omega}$ é a direção de saída da radiância,

f_r é a *BRDF*,
L_i é a radiância incidente na posição x,
$(\vec{n}_x \cdot \vec{\omega}')$ é uma atenuação proporcional ao ângulo incidente,
$d\vec{\omega}'$ é o ângulo sólido,
Ω é o hemisfério acima da posição x.

Para solucionar essa integral, é necessária a informação sobre a radiância incidente L_i. O *photon map*, provê a informação sobre o fluxo incidente Φ_i, também chamado irradiância. Utilizando a relação entre radiância e irradiância, o termo L_i pode ser reescrito da seguinte maneira:

$$L_i(x, \vec{\omega}') = \frac{d^2\Phi_i(x, \vec{\omega}')}{(\vec{n}_x \cdot \vec{\omega}')d\vec{\omega}' dA_i} \tag{4.2}$$

reescrevendo então a integral,

$$\begin{aligned} L_r(x, \vec{\omega}) &= \int_{\Omega_\infty} f_r(x, \vec{\omega}, \vec{\omega}') \frac{d^2\Phi_i(x, \vec{\omega}')}{(\vec{n}_x \cdot \vec{\omega}')d\vec{\omega}' dA_i} (\vec{n}_x \cdot \vec{\omega}')d\vec{\omega}' \\ &= \int_{\Omega_\infty} f_r(x, \vec{\omega}, \vec{\omega}') \frac{d^2\Phi_i(x, \vec{\omega}')}{dA_i}. \end{aligned}$$

Contextualizando, a irradiância Φ_i pode ser aproximada através do somatório da energia $\triangle\Phi_p(\vec{\omega}')$ dos n fótons mais próximos da posição x. Logo, o cálculo da radiância refletida pode ser definido como:

$$L_r(x, \vec{\omega}) \approx \sum_{p=1}^{n} f_r(x, \vec{\omega}, \vec{\omega}') \frac{\Delta\Phi_i(x, \vec{\omega}')}{\Delta A} \tag{4.3}$$

Assumindo que a superfície é plana em torno de x, e que a busca pelos fótons pode ser ilustrada como uma esfera com centro em x, sendo expandida até conter n fótons; o termo $\triangle A$ pode ser encontrado projetando a esfera na superfície e computando a área do círculo resultante. Logo:

$$\Delta A = \pi r^2 \tag{4.4}$$

sendo r o raio da esfera, definido pela maior distância entre x e os n fótons encontrados. Então, a equação que define o cálculo da radiância refletida em uma superfície utilizando *photon map* é:

$$L_r(x, \vec{\omega}) \approx \frac{1}{\pi r^2} \sum_{p=1}^{n} f_r(x, \vec{\omega}, \vec{\omega}') \Delta\Phi_i(x, \vec{\omega}'). \tag{4.5}$$

O algoritmo 4.8 ilustra o cálculo da radiância, referente a equação 4.5, para um

```
CalcularRadiancia(x, ω⃗, n⃗)
  localizar os n fótons mais próximos;
  r ⇐ distância ao n-ésimo fóton mais próximo;
  ∑ flux ⇐ 0;
  Foreach fóton p Do
     p⃗_d ⇐ direção do fóton;
     Φ_p ⇐ energia do fóton;
     ∑ flux+ = f_r(x, ω⃗', p⃗_d) * Φ_p
  L_r ⇐ ∑ flux (πr²);
  Return L_r;
End CalcularRadiancia
```

Figura 4.8: Cálculo da radiância

raio $\vec{\omega}$ incidente em um ponto x com normal $\vec{n}$.

Filtros

Jensen [13] propõe o uso de filtros ao se calcular a radiância utilizando *photon mapping*, principalmente em casos onde poucos fótons são utilizados, evitando efeitos não desejados na renderização. Entre os filtros propostos está o filtro cone, implementado e utilizado na geração de todos os resultados presentes no capítulo 7.

O filtro cone atribui um peso ω_{pc} a cada fóton, baseado na distância d_p entre o ponto x onde a radiância está sendo avaliada e o fóton p. O peso é calculado por

$$\omega_{pc} = 1 - \frac{d_p}{kr}, \tag{4.6}$$

onde $k \geq 1$ é uma constante que caracteriza o filtro e r é a distância máxima.

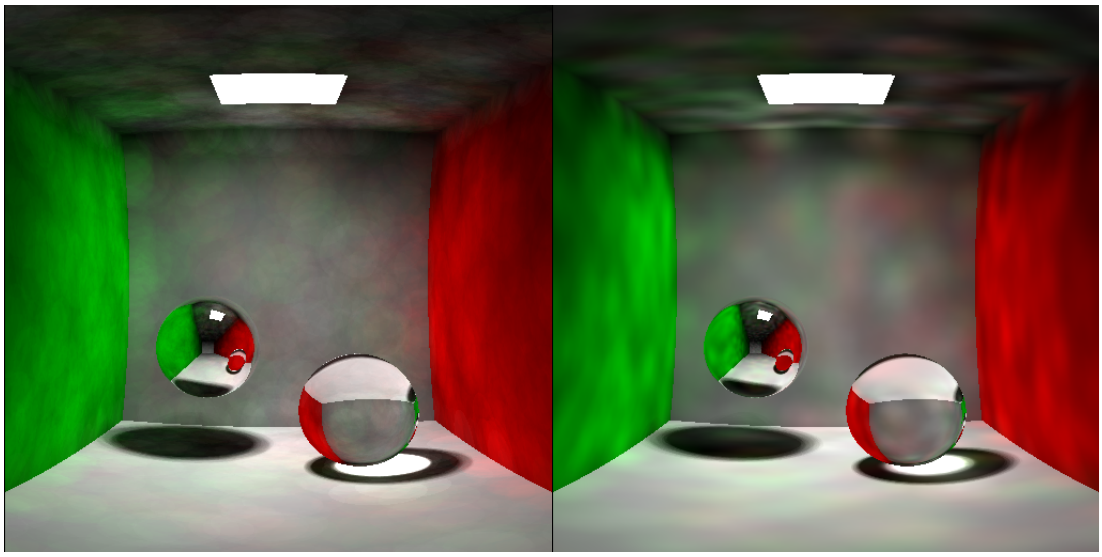
Considerando que os fótons considerados no cálculo da radiância estão em uma superfície plana, utiliza-se uma normalização para distribuição dos fótons igual a $1 - \frac{2}{3k}$. O cálculo da radiância utilizando o filtro é

$$L_r(x, \vec{\omega}) \approx \frac{\sum_{p=1}^{N} f_r(x, \vec{\omega}_p, \vec{\omega}) \Delta\Phi_p(x, \vec{\omega}_p) \omega_{pc}}{(1 - \frac{2}{3k})\pi r^2}. \tag{4.7}$$

A figura 4.9 mostra uma comparação entre a renderização de uma cena com e sem o uso do filtro cone.

4.3.2 Renderização utilizando Photon Map

Entendido o processo que permite a recuperação da informação da radiância, torna-se possível a renderização utilizando o *photon map*. Após o computo de todos os



(a) (b)

Figura 4.9: Comparação entre cenas renderizadas (a) sem filtro cone e (b) com filtro cone.

passos descritos nas seções anteriores, o processo final de renderização é simples. Utilizando um algoritmo de *raytracing*, raios são traçados pela cena a partir do observador para recuperar a informação de cor de cada pixel da imagem final. Em seu livro, Jensen descreve o processo matematicamente, através da decomposição da equação de renderização, e da explicação de como cada termo advindo dessa decomposição é computado utilizando-se *raytracing* e *photon mapping*.

O *photon map* não armazena informações sobre a iluminação direta, ou seja, a primeira interação de cada fóton após ser lançado a partir da fonte de luz não é considerada. Como geralmente a radiância em uma cena é decidida predominantemente pela iluminação direta, o seu cálculo é feito como no algoritmo clássico de *raytracing*, com o objetivo de prover precisão ao resultado final. O primeiro passo é verificar quais fontes de luz influenciam na iluminação de um determinado ponto. Para isso são lançados *shadow rays*, ou raios de sombra, com origem no ponto em direção a fonte de luz para verificar a visibilidade entre ambos. Para uma fonte de luz pontual, apenas um *shadow ray* é lançado, enquanto que para uma luz quadrangular é necessário realizar uma amostragem utilizando diversos raios, pois a luz é emitida ao longo de uma superfície. A contribuição final de uma luz quadrangular é dada pela média das contribuições de cada *shadow ray* lançado.

Como explicado na subseção 4.1.3, a interação dos fótons com superfícies especulares não são armazenadas. Para avaliar a radiância em um ponto pertencente a uma superfície especular, é utilizado o princípio do *Monte Carlo raytracing*, então vários raios de reflexão são gerados integrando o valor da iluminação retornado por cada um deles. Em uma superfície difusa, o valor da radiância é determinado

pela contribuição da iluminação direta em um dado ponto, juntamente com as informações referentes ao cálculo da irradiância fornecido pelo *photon mapping*, que provê informações referentes a outros efeitos de iluminação global, como por exemplo, cáusticas. O resultado final é obtido ao se integrar os resultados dos processos de avaliação da iluminação descritos.

Capítulo 5

Modelos de Iluminação

A luz emitida em uma cena, interage com os objetos presentes de diferentes maneiras, e a aparência dos materiais diferem em condições de iluminação semelhantes. A luz incidente em uma superfície pode ser refletida, transmitida ou absorvida, de acordo com as propriedades de cada material. Alguns modelos desenvolvidos para descrever essa interação e implementados neste trabalho são apresentados neste capítulo.

5.1 Lambertiano

Apesar de não ser fisicamente plausível, é uma boa aproximação para muitas superfícies reais. O modelo descreve uma superfície perfeitamente difusa, que espalha a iluminação incidente igualmente em todas as direções, como ilustrado na Figura 5.1. Logo, sua BRDF é constante para qualquer $\vec{\omega}$:

$$f_r = \frac{\rho_d}{\pi}, \tag{5.1}$$

onde ρ representa a reflectância da superfície, ou seja, a fração de energia incidente que é refletida pela superfície.

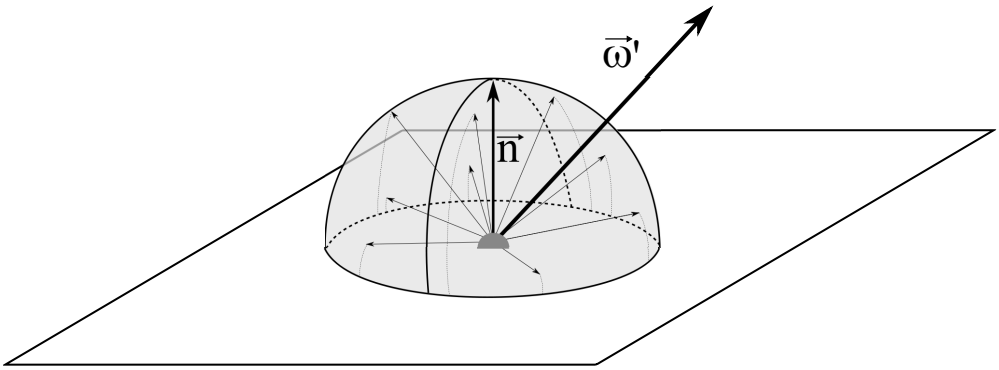


Figura 5.1: Reflexão Lambertiana

5.2 Reflexão e Transmissão Especular

O comportamento da luz ao interagir com superfícies perfeitamente lisas é relativamente fácil de ser caracterizado. Nesta seção são apresentados os cálculos da direção de reflexão ou transmissão da luz incidente, e da fração de luz refletida ou transmitida pelo ambiente, utilizando as equações de Fresnel.

5.2.1 Equações de Fresnel

Geralmente em algoritmos simples de *raytracing*, a quantidade de luz refletida ou transmitida é determinada por fatores de refletividade ou transmissividade, constantes em toda uma superfície. Essa quantidade é expressa de maneira física pelas equações de *Fresnel*.

Existem dois grupos de equações de *Fresnel*: para meios dielétricos (que não conduzem eletricidade, como o vidro) e para meios condutores (como metais). Em ambos os casos as equações tem variações dependendo da polarização (paralela ou perpendicular) da luz incidente, porém, devido a complexidade de considerar essa polarização, geralmente assume-se em algoritmos de renderização (inclusive neste trabalho) que a luz não é polarizada. Dada essa simplificação, a reflectância de *Fresnel* é calculada pela média dos termos de polarização perpendicular e paralelo elevados ao quadrado (equação 5.4).

Para computar a reflectância de *Fresnel* para um meio dielétrico, é necessário o valor dos índices de refração do meio por onde a luz propaga ao incidir em uma superfície, e do meio referente ao material que compõe a superfície em questão. Valores dos índices de refração para diversos tipos de materiais podem ser encontrados na literatura. Alguns índices de materiais comumente utilizados são: ar, $\eta \approx 1.0$; água, $\eta \approx 1.33$; e vidro, $\eta \approx [1.5, 1.7]$, dependendo do tipo de material. O cálculo da reflectância de *Fresnel* para meios dielétricos com a luz com polarização paralela e perpendicular é

$$\begin{aligned} \rho_{\parallel} &= \frac{\eta \cos\theta' - \eta' \cos\theta}{\eta \cos\theta' + \eta' \cos\theta}, \\ \rho_{\perp} &= \frac{\eta' \cos\theta' - \eta \cos\theta}{\eta' \cos\theta' + \eta \cos\theta} \end{aligned} \tag{5.2}$$

onde $\rho_{\parallel}$ é a reflectância de *Fresnel* para a luz com polarização paralela, $\rho_{\perp}$ é a reflectância de *Fresnel* para luz com polarização perpendicular, η' e η são os índices de refração dos meios incidente e transmitido.

Diferentemente dos meios dielétricos, os meios condutores não transmitem luz, porém uma parcela da luz incidente é absorvida pelo material. A equação de *Fresnel*

para meios condutores determina o quanto de luz é refletida, e é definida como

$$\begin{aligned} \rho_{\parallel}^2 &= \frac{(\eta^2+k^2)\cos\theta'^2 - 2\eta\cos\theta' + 1}{(\eta^2+k^2)\cos\theta'^2 + 2\eta\cos\theta' + 1} \\ \rho_{\perp}^2 &= \frac{(\eta^2+k^2) - 2\eta\cos\theta' + \cos\theta'^2}{(\eta^2+k^2) + 2\eta\cos\theta' + \cos\theta'^2} \end{aligned} \tag{5.3}$$

onde η é o índice de refração do condutor e k é um coeficiente de absorção. Alguns exemplos de valores para materiais condutores são: ouro, $\eta \approx 0.370, k \approx 2.820$; prata, $\eta \approx 0.177, k \approx 3.638$; e aço, $\eta \approx 2.485, k \approx 3.433$.

Finalmente, a reflectância de *Fresnel* para luzes não polarizadas, e utilizada neste trabalho é dada por

$$F_r = \frac{1}{2}(\rho_{\parallel}^2 + \rho_{\perp}^2). \tag{5.4}$$

5.2.2 Reflexão Especular

Reflexões especulares acontecem quando a luz incide em superfícies lisas, podendo ser refletida de forma perfeitamente especular ou de forma *glossy*, que acontece quando a superfície contém alguma rugosidade ou imperfeição. Em superfícies perfeitamente lisas a reflexão se comporta como em um espelho perfeito, onde a direção da luz incidente $\vec{\omega}'$ e da luz refletida $\vec{\omega}$ fazem o mesmo ângulo com a normal $\vec{n}$ da superfície (Figura 5.2), logo:

$$\vec{\omega} = 2(\vec{\omega}' \cdot \vec{n})\vec{n} - \vec{\omega}'. \tag{5.5}$$

A BRDF para a reflexão especular perfeita pode ser expressa utilizando coordenadas esféricas para a direção,

$$f_r(x, \vec{\omega}', \vec{\omega}) = 2\rho_s\delta(\sin^2\theta' - \sin^2\theta)\delta(\phi' - \phi \pm \pi), \tag{5.6}$$

onde $\vec{\omega} = (\theta, \phi)$, $\vec{\omega}' = (\theta', \phi')$ e a função delta de *Dirac*, $\delta(x)$, é usada para limitar a direção em que a BRDF não é zero (quando $x = 0$).

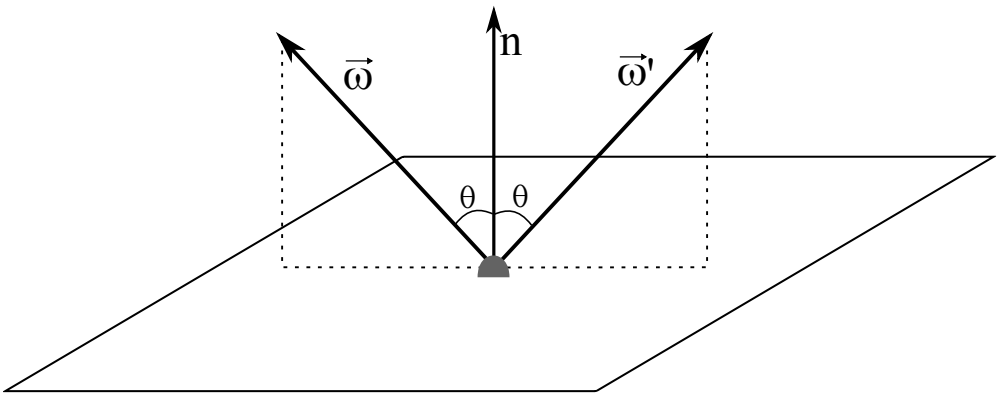


Figura 5.2: Reflexão Especular Perfeita

Porém, como demonstrado por Pharr e Humphreys [1], essa definição contêm um erro, pois ao ser aplicada no cálculo da radiância refletida gera um termo $\cos\theta^{'}$ extra. É definida então uma nova formulação,

$$f_r(x, \vec{\omega}^{'}, \vec{\omega}) = F_r(\vec{\omega^{'}}) \frac{\delta(\vec{\omega} - R(\vec{\omega}^{'}, \vec{n}))}{|\cos\theta|} \tag{5.7}$$

onde $R(\vec{\omega}^{'}, \vec{n})$ é um vetor de reflexão especular em relação a normal $\vec{n}$.

5.2.3 Transmissão Especular

Considerando uma radiância incidente em uma superfície, que determina a divisão entre dois meios com índices de refração diferentes $\eta^{'}$ e η para os meios de entrada e saída da luz, respectivamente, pode-se calcular a direção da luz transmitida baseando-se na lei de *Snell*,

$$\eta^{'} \sin\theta^{'} = \eta \sin\theta. \tag{5.8}$$

A lei pode ser usada diretamente para cálculo da direção da luz transmitida, e utilizada na derivação da BTDF que define a transmissão especular,

$$f_t(x, \vec{\omega}^{'}, \vec{\omega}) = \frac{\eta^2}{\eta^{'2}}(1 - F_r(\vec{\omega})) \frac{\delta(\vec{\omega} - T(\vec{\omega}^{'}, \vec{n}))}{|\cos\theta|}, \tag{5.9}$$

onde $T(\vec{\omega}^{'}, \vec{n})$ é um vetor de transmissão especular em relação a normal $\vec{n}$.

5.3 Modelos de Microfacetas

Nesses modelos, desenvolvidos com objetivo de tratar a iluminação em superfícies rugosas, as superfícies são modeladas como um conjunto de pequenas microfacetas. Uma superfície composta de microfacetas é essencialmente um mapa de alturas onde a distribuição das faces é descrita estatisticamente. Duas distribuições representando níveis de rugosidade diferentes são ilustradas na Figura 5.3.

Esse modelo permite uma série de fenômenos: uma microfaceta pode ser ocluída por outra, pode provocar sombras em uma microfaceta próxima e também podem ocorrer inter-reflexões entre microfacetas.

Nas seções 5.4 e 5.5 são descritos modelos de iluminação de microfacetas difusas e especulares respectivamente.

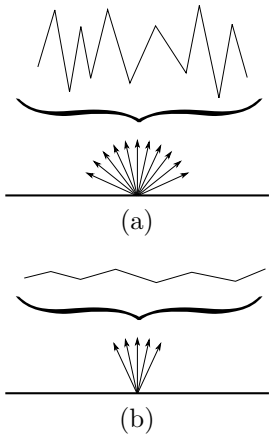


Figura 5.3: Representação do modelo de microfacetas, onde a rugosidade é representada pela distribuição das normais da superfície; (a) superfície muito rugosa e (b) superfície pouco rugosa, i.e., com pouca variação na distribuição.

5.4 Oren-Nayar

Apesar do modelo lambertiano ser um dos mais usados na computação gráfica, é inadequado para simular superfícies rugosas. Enquanto no modelo lambertiano a superfície tem o aspecto igualmente iluminado em todas as direções, em uma superfície rugosa a iluminação depende do ponto em que está sendo visualizada. No modelo Oren-Nayar [24] o brilho da superfície aumenta a medida que a direção do observador se aproxima da direção da fonte de luz.

A superfície é modelada como uma coleção de microfacetas perfeitamente lambertianas e como um modelo de microfacetas, efeitos como sombreamento, oclusão, e inter-reflexão entre elas são simulados. O cálculo da distribuição das microfacetas é dado por uma distribuição gaussiana com um fator σ que determina o desvio padrão do ângulo de orientação. A função que define o modelo é:

$$f_r(x, \vec{\omega}', \vec{\omega}) = \frac{\rho}{\pi}(A + B \max(0, \cos((\phi_i - \phi_o))) \sin\alpha \tan\beta) \tag{5.10}$$

onde

$$A = 1 - \frac{\sigma^2}{2(\sigma^2 + 0.33)},$$

$$B = \frac{0.45\sigma^2}{\sigma^2 + 0.09},$$

$$\alpha = \max(\theta_i, \theta_o),$$

$$\beta = \min(\theta_i, \theta_o),$$

e σ é medido em radianos.

5.5 Torrance-Sparrow

No modelo especular perfeito, a luz incidente é refletida em uma única direção, no entanto, no mundo real não existem materiais com tal característica. Torrance e Sparrow [8] desenvolveram um modelo para descrever reflexões especulares em superfícies rugosas. Neste modelo, superfícies são modeladas como coleções de microfacetas especulares.

Este modelo insere um termo de atenuação geométrica $G(\vec{\omega}, \vec{\omega}')$, que descreve a fração de microfacetas que são oclusas ou sombreadas. Dada as direções $\vec{\omega}'$ e $\vec{\omega}$, é calculado da seguinte maneira:

$$G(\vec{\omega}, \vec{\omega}') = \min(1, \min(\alpha(n \cdot \vec{\omega}), \alpha(n \cdot \vec{\omega}'))). \tag{5.11}$$

onde ω_h é a direção da normal da microfaceta, definida pelo ângulo médio entre $\vec{\omega}'$ e $\vec{\omega}'$, e

$$\alpha = \frac{2(n \cdot \omega_h)}{\vec{\omega} \cdot \omega_h}. \tag{5.12}$$

As microfacetas individualmente refletem a luz de acordo com a lei de Fresnel $F_r(\vec{\omega})$, ponderada por uma função estatística de distribuição $D(\omega_h)$ computada na normal da microfaceta. O cálculo da BRDF é dado por:

$$f_r(p, \vec{\omega}, \vec{\omega}') = \frac{D(\omega_h)G(\omega_h, \vec{\omega}')F_r(\vec{\omega})}{4\cos(\theta_o)\cos(\theta_i)}. \tag{5.13}$$

No trabalho original Torrance e Sparrow usaram uma função Gaussiana de distribuição, porém variações que estendem o modelo usando outras funções de distribuição foram desenvolvidas. As subseções 5.5.1 e 5.5.2 apresentam as que foram implementadas neste trabalho.

5.5.1 Blinn

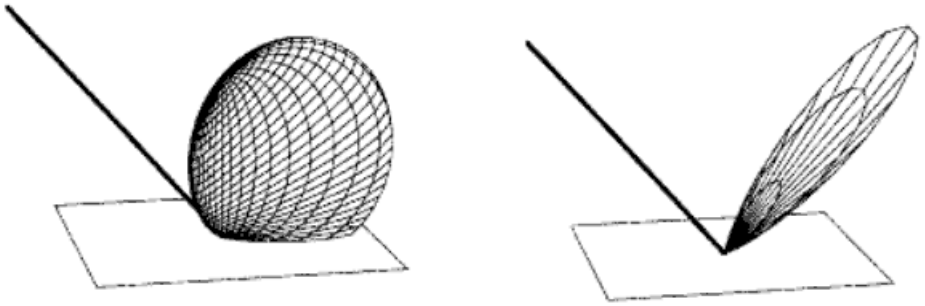
Na função de distribuição de Blinn [9] as normais das microfacetas são aproximadas por uma variação exponencial, controlada por um expoente e que determina se essa variação acontecerá de forma mais gradual, ou mais brusca. A função de distribuição proposta por Blinn é dada por:

$$D(\omega_h) = \frac{e+2}{2\pi}(\omega_h \cdot n)^e \tag{5.14}$$

onde o expoente e determina a probabilidade da direção das normais das microfacetas se distanciarem da direção da normal da superfície, determinando a rugosidade.

A figura 5.4 mostra como o expoente e afeta a distribuição. Com o valor de e igual a 4, como mostrado na figura 5.4a, existe uma probabilidade maior da direção

das normais das microfacetas se distanciarem da direção da normal da superfície, o que caracteriza uma superfície áspera. Na figura 5.4b, com e igual a 20, as normais das microfacetas se aproximam da normal da superfície, então a reflexão se aproxima do comportamento especular perfeito, caracterizando uma superfície lisa.



(a) $e = 4$ (b) $e = 20$

Figura 5.4: Representação da reflexão da luz utilizando o modelo de Torrance-Sparrow com função de distribuição de Blinn com diferentes valores de e. Figura retirada do livro *Physically Based Rendering* [1]

A distribuição de Blinn é isotrópica, pois depende somente do ângulo entre o vetor médio (normal da microfaceta) e a normal da superfície. Na prática significa que se girarmos um determinado ponto da superfície em torno de sua normal, a quantidade de luz refletida será sempre a mesma.

5.5.2 Anisotrópico

Ashikhmin e Shirley [25] desenvolveram uma função de distribuição para modelar superfícies anisotrópicas. Diferentemente das superfícies isotrópicas, nas superfícies anisotrópicas a forma como a luz é refletida em um determinado ponto, varia se o objeto for rotacionado em torno da normal da superfície neste ponto. Exemplos de superfícies com tais características são: metais escovados, *cd-roms* e superfícies aveludadas, entre outras.

Para o cálculo da função de distribuição anisotrópica são utilizados dois parâmetros e_x e e_y (figura 5.5), que definem a rugosidade da superfície. O parâmetro e_x define um expoente para a função de distribuição que determina uma tendência de orientação da normal da microfaceta ao longo do eixo x, e e_y ao longo do eixo y. A função de distribuição anisotrópica desenvolvida por Ashikhmin e Shirley é dada por:

$$D(\omega_h) = \frac{\sqrt{(e_x + 2)(e_y + 2)}}{2\pi}(\omega_h \cdot n)^{e_x \cos^2 \phi + e_y \sin^2 \phi}. \tag{5.15}$$

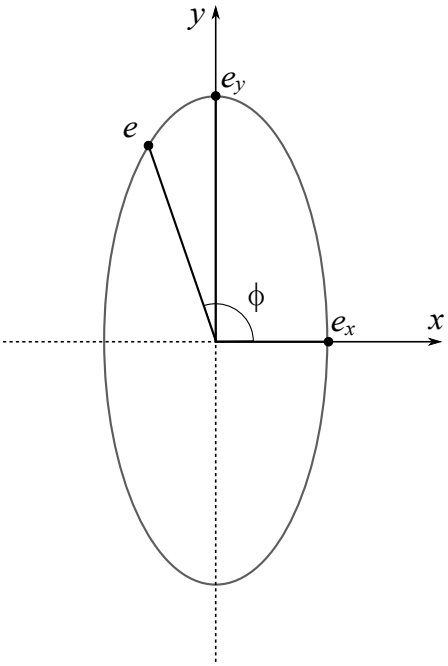


Figura 5.5: Representação dos parâmetros e_x e e_y da função de distribuição anisotrópica.

5.6 Schlick

O modelo apresentado por Schlick [26] é o mais elaborado entre os apresentados neste trabalho. Não descreve um material com apenas uma característica (e.g. difuso ou especular), mas define uma BRDF que descreve materiais com comportamento tanto difuso, *glossy*, ou especular, determinado pelos parâmetros que o definem.

O modelo de Schlick contem três principais características. Primeiro, apesar de simples e intuitivo, obedece as principais leis da física para iluminação (lei da conservação da energia, lei da reciprocidade de Helmholtz, teoria das microfacetas, equação de Fresnel). Segundo, é definido por um número pequeno de parâmetros que podem ser definidos intuitivamente ou baseados em medições experimentais. Terceiro, é expresso por uma formulação com complexidade variável, dependendo dos fenômenos físicos que pretendem ser incluídos (reflexão isotrópica ou anisotrópica, auto-sombreamento, material homogêneo ou heterogêneo).

Os parâmetros que definem o modelo são:

- $F_0 \in [0, 1]$, fator de reflexão especular.
- $\sigma \in [0, 1]$, fator de rugosidade ($\sigma = 0$ é perfeitamente liso e especular, $\sigma = 1$ muito rugoso e lambertiano).
- $\psi \in [0, 1]$, fator de isotropia ($\psi = 0$ é perfeitamente anisotrópico, $\psi = 1$ é isotrópico).

A BRDF de Schilick é um combinação entre um fator especular $S(u)$ e termos para controlar a quantidade de reflexão difusa, *glossy* e especular (d, g, s respectivamente).

$$f_r(x, \vec{\omega}, \vec{\omega}') = S(u) \left\{ \frac{d}{\pi} + gD(t, v, v', w) + sf_{r,s}(x, \vec{\omega}, \vec{\omega}') \right\}, \tag{5.16}$$

onde $f_{r,s}$ é a BRDF especular perfeita e $D(t, v, v', w)$ é um termo de controle direcional da reflexão *glossy*. Os parâmetros u, t, v, v' e w são computados a partir da normal da superfície $\vec{n}$, da direção da luz incidente $\vec{\omega}'$ e da luz refletida $\vec{\omega}$:

$$\begin{aligned}
\vec{H} &= \frac{\vec{\omega} + \vec{\omega}'}{\|\vec{\omega} + \vec{\omega}'\|} \\
u &= \vec{\omega} \cdot \vec{H} \\
t &= \vec{n} \cdot \vec{H} \\
v &= \vec{\omega} \cdot \vec{n} \\
v' &= \vec{\omega}' \cdot \vec{H} \\
w &= \vec{T} \cdot \frac{\vec{H} - (\vec{n} \cdot \vec{H})\vec{n}}{|\vec{H} - (\vec{n} \cdot \vec{H})\vec{n}|}
\end{aligned}$$

como ilustrado na Figura 5.6.

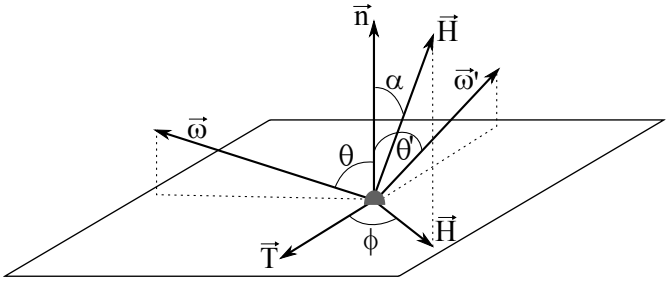


Figura 5.6: Definições do modelo de Schlick.

O fator de reflexão especular $S(u)$ é dado pela aproximação de Fresnel,

$$S(u) = F_0 + (1 - F_0)(1 - u)^5. \tag{5.17}$$

Os termos d, g e s da equação 5.16, controlam as reflexões difusa, *glossy* e especular baseados no fator de rugosidade σ,

$$g = 4\sigma(1 - \sigma), \tag{5.18}$$

$$d = \begin{cases} 0 & \text{para} \sigma < 0.5 \\ 1 - g & \text{caso contrário,} \end{cases} \tag{5.19}$$

$$s = \begin{cases} 1 - g & \text{para} \sigma < 0.5 \\ 0 & \text{caso contrário.} \end{cases} \tag{5.20}$$

Como dito, o modelo obedece a teoria de microfacetas. A função de distribuição $D(t, v, v^{'}, w)$ determina a orientação das microfacetas através de dois termos $Z(t)$ e $A(w)$ e de uma restrição geométrica $G(v)$ das microfacetas. Esses termos são definidos como:

$$Z(t) = \frac{\sigma}{(1 + \sigma t^2 - t^2)^2}, \tag{5.21}$$

$$A(w) = \sqrt{\frac{\psi}{\psi^2 + \psi^2 w^2 - w^2}}, \tag{5.22}$$

$$G(v) = \frac{v}{\sigma - \sigma v + v}. \tag{5.23}$$

Logo, a função de distribuição $D(t, v, v^{'}, w)$ é

$$D(t, v, v^{'}, w) = \frac{G(v)G(v^{'})Z(t)A(w)}{4\pi v v^{'}} + \frac{1 - G(v)G(v^{'}}{\pi} A(w). \tag{5.24}$$

Definidos os modelos de iluminação utilizados neste trabalho, torna-se possível então a análise dos resultados gerados por cada um deles, o que é feito no capítulo 7. Porém antes, no capítulo 6, detalhes da implementação são descritos e discutidos.

Capítulo 6

Implementação

Como dito, uma das propostas do trabalho é o entendimento prático do método aliado a parte teórica apresentada, possibilitando uma elucidação mais ampla do tema. Ao longo de todo o trabalho, além dos conceitos teóricos são apresentados algoritmos e estruturas de dados utilizados, e esse capítulo tem por objetivo apresentar algumas especificidades da implementação assim como o impacto delas nos resultados finais.

O programa foi implementando com a linguagem C++ e utilizando o paradigma de orientação a objetos. Como ficará evidente neste capítulo, essa escolha permitiu o desenvolvimento de um código estruturado onde futuras extensões possam ser realizadas de forma prática e rápida. Foi utilizado também a biblioteca OpenGL [27] para a renderização dos resultados e a biblioteca GLUT [28] para a interface com o usuário e gerenciamento de janelas.

6.1 Fontes de Luz

As fontes de luz implementadas no trabalho foram as do tipo pontual e quadrangular, porém, como dito na subseção 4.1.1, existem diversos tipos de fontes de luz além desses. Com o objetivo de facilitar a inclusão de outros tipos, foi criada uma classe `Light`, contendo as informações comuns entre os tipos de fontes de luz, como cor e energia, além das funções virtuais `generatePhotonRay` e `getIrradiance` que devem ser implementadas pelas classes filhas, de acordo com as suas especificidades. A função `generatePhotonRay` tem o objetivo de gerar um fóton com origem na fonte de luz, e é utilizada no passo de *photon tracing*. A `getIrradiance` retorna a irradiância referente ao ponto de luz, em um determinado ponto de uma superfície (representado pelo parâmetro `intersection_data`) e é utilizada no passo de renderização. O diagrama na figura 6.1 mostra o relacionamento entre as classes.

A classe `Light` herda da classe `Intersectable`, pois como o objetivo dos algoritmos de iluminação global é o realismo, as fontes de luz também tem uma repre-

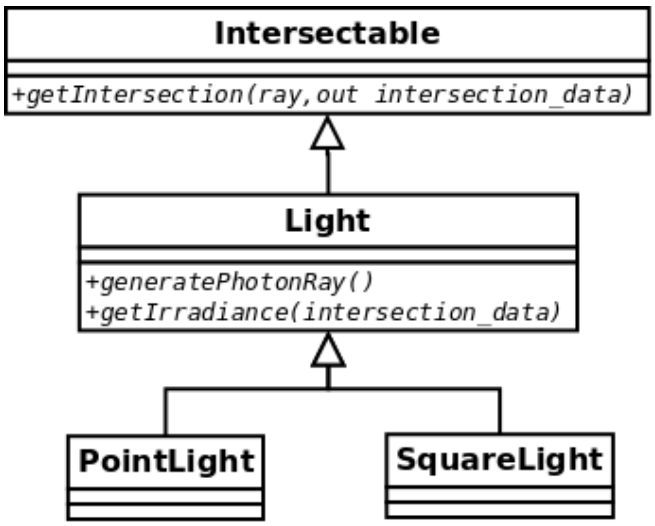


Figura 6.1: Diagrama de Classes - Fontes de Luz

sentação visual na cena, que é computada através do traçado de raios e verificação das interseções com os objetos presentes.

6.1.1 Luz Quadrangular e Sombras

Como apresentado no capítulo 4, o *photon map* armazena somente informações sobre iluminação indireta, sendo a iluminação direta de uma fonte de luz em um determinado ponto calculada da mesma forma que no algoritmo clássico de *raytracing*. Para decidir se em determinado ponto a luz direta proveniente de uma fonte de luz contribui para o cálculo da radiância incidente é necessário testar a visibilidade entre este ponto e a fonte de luz em questão. Para uma luz pontual o teste é simples, sendo necessário somente um raio para verificar se algum objeto da cena impossibilita a incidência da luz. Porém, em uma fonte de luz quadrangular (e também em outros tipo de fontes que emitem a luz ao longo de uma superfície) esse teste é feito utilizando-se diversos raios (amostras) para se obter um percentual da luz que incide em um ponto de uma superfície, como mostrado na Figura 6.2.

Na implementação deste trabalho a classe que representa a fonte de luz quadrada contém uma constante `NUM_SHADOW_SAMPLES`, que indica o número de amostras a serem utilizadas nos testes de iluminação direta. O valor de `NUM_SHADOW_SAMPLES` influi de forma direta no resultado final do algoritmo como pode ser visto na Figura 6.3. Como pode-se perceber, o aumento da quantidade de amostras melhora o resultado, tornando a região de penumbra mais suave.

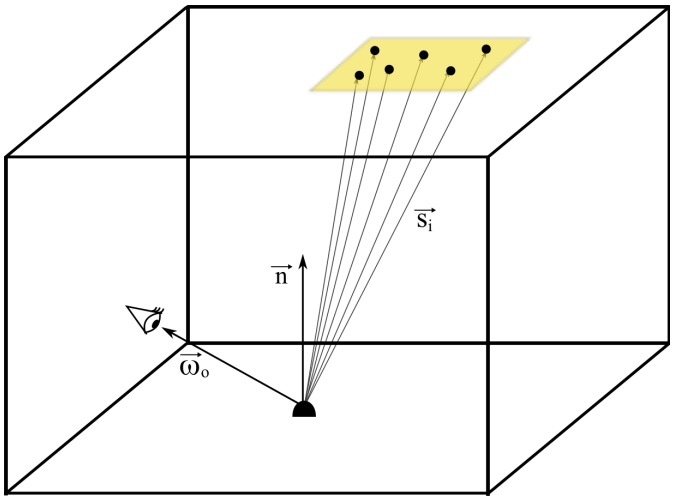


Figura 6.2: Iluminação direta com uma luz quadrangular. A partir do ponto de interseção são lançados raios de sombra s_i em direção à luz. Neste caso apenas seis raios são utilizados para efeitos ilustrativos. A a intensidade da sombra no ponto é definida pela quantidade de raios não bloqueados, ou seja, aqueles que atingem primeiro a fonte de luz antes de qualquer outro objeto.

6.2 Objetos

Assim como as classes de luz, os objetos (representados pela classe `Object3D`) também herdam de `Intersectable`, permitindo a inclusão de diversos tipos de objetos geométricos. Neste trabalho foram implementados alguns objetos triviais, como planos infinitos (que representam as paredes dos cenários nos exemplos expostos), esferas e caixas, além de uma classe para tratar malhas triangulares.

Para realizar as interseções dos raios com os triângulos da malha foram utilizadas coordenadas baricêntricas $(\lambda_0, \lambda_1, \lambda_2)$. Assumindo um sistema local de coordenadas em relação à face definida pelos vértices $\{v_0, v_1, v_2\}$, qualquer ponto p pode ser expresso em coordenadas baricêntricas, isto é, por uma combinação de pesos dos vértices do triângulo:

$$p = \lambda_0 v_0 + \lambda_1 v_1 + \lambda_2 v_2, \tag{6.1}$$

onde, $\lambda_0 + \lambda_1 + \lambda_2 = 1$. Em forma matricial temos então $T \cdot \lambda = p$, ou explicitamente:

$$\begin{bmatrix} x_0 & x_1 & x_2 \\ y_0 & y_1 & y_2 \\ 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} \lambda_0 \\ \lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix} \tag{6.2}$$

Para encontrar o vetor desconhecido λ basta inverter a matriz:

$$\lambda = T^{-1} \cdot p. \tag{6.3}$$

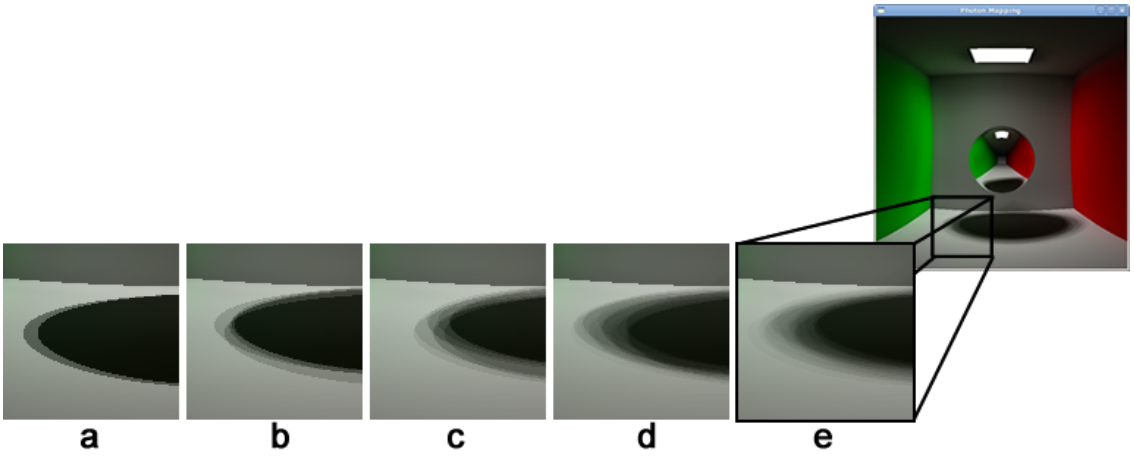


Figura 6.3: Exemplos de sombras criadas com diferentes números de amostras (*shadow rays*). (a) 2 amostras; (b) 5 amostras; (c) 10 amostras; (d) 25 amostras; (e) 50 amostras.

Porém, sem perda de generalidade, podemos definir um sistema local de coordenadas onde $v_0 = (0, 0)$, $v_1 = (x_1, 0)$ e $v_2 = (x_2, y_2)$, como ilustrado na Figura 6.4. Desta forma, a matriz T é simplificada da seguinte maneira:

$$\begin{bmatrix} 0 & x_1 & x_2 \\ 0 & 0 & y_2 \\ 1 & 1 & 1 \end{bmatrix} \tag{6.4}$$

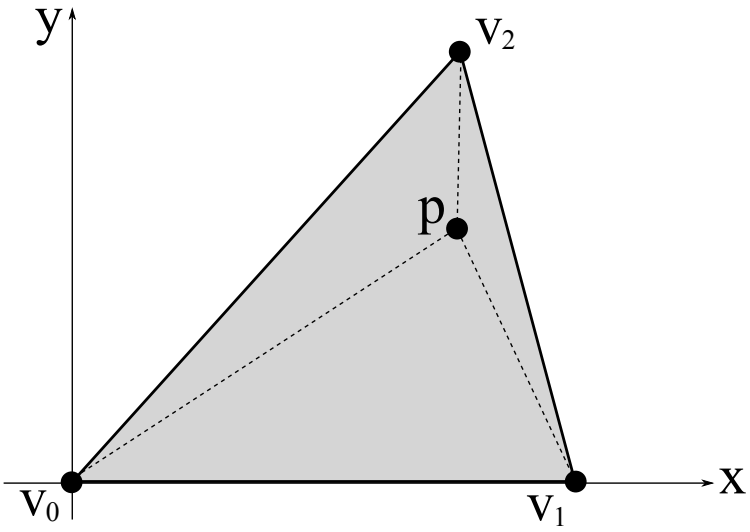


Figura 6.4: Triângulo em coordenadas locais.

Estes três valores x_1, x_2, y_2 são armazenados por face, assim a matriz inversa e o determinante podem ser rapidamente calculados. Da mesma forma, os dois vetores tangentes à face usados para realizar a transformação para coordenadas locais são armazenados.

Uma vantagem em se trabalhar com coordenadas baricêntricas é que um ponto

pode ser facilmente testado para saber se está dentro ou fora do triângulo, basta para tanto verificar se cada componente de λ está no intervalo $[0, 1]$. Este teste é utilizado exaustivamente, por exemplo, para verificar a interseção raio-triângulo: primeiro é encontrado o ponto onde o raio intersecta o plano do triângulo, e em seguida é realizado o teste para saber se este ponto está contido no próprio. Ainda mais, as coordenadas baricêntricas podem ser igualmente utilizadas para interpolar as normais dentro de um triângulo. Note que para tanto, as normais por vértice da malha são previamente calculadas.

6.3 Modelos de Iluminação

Para a implementação dos modelos de iluminação foi utilizada uma estrutura de classes que permite que outros modelos sejam facilmente incluídos no programa. Como pode ser visto na Figura 6.5, foi criada uma classe abstrata para representar o comportamento de uma BSDF, e os modelos implementados são filhos dessa classe. Na implementação, o conceito de BSDF foi estendido em relação a sua definição teórica. Conceitualmente, como dito na subseção 2.2.1, uma BSDF determina a quantidade de luz refletida ou transmitida, dado as direções de chegada e saída da luz. Esse valor é determinado pela função `f` que recebe essas informações como parâmetros, além de uma normal, necessária para o cálculo da função em alguns dos modelos. A extensão citada está relacionada a implementação da função `sample_f`, que a partir de uma direção de chegada da luz, calcula, respeitando as características do modelo que representa, uma direção provável de saída da luz, além de retornar o valor da BSDF para essa direção.

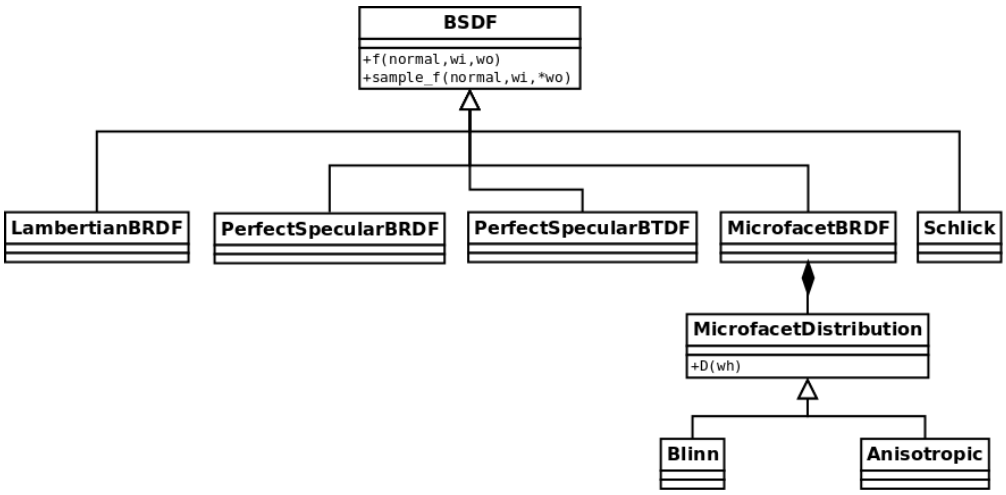


Figura 6.5: Diagrama de Classes - Modelos de Iluminação

A classe `Microfacet` é uma generalização do modelo de *Torrance-Sparrow* apresentado na seção 5.5. Como demonstrado, o modelo pode ser estendido através

da utilização de diferentes funções de distribuição; neste trabalho foram implementadas as de *Blinn* e a *Anisotrópica*. Para representar essa relação e permitir que novas funções de distribuição sejam inseridas facilmente, foi criada uma classe abstrata `MicrofacetDistribution` que se relaciona com a classe `MicrofacetBRDF` por composição, e é implementada pelas classes `Blinn` e `Anisotropic`.

6.3.1 Amostragem na Reflexão

Para se estimar a radiância em uma superfície especular, como dito no Capítulo 4, não é utilizado as informações do *photon map*. O cálculo é feito, assim como nos algoritmos de *Monte Carlo raytracing*, através da geração de diversos raios de reflexão para se estimar a quantidade de luz incidente em um ponto de uma superfície especular. A quantidade de raios de reflexão é controlada no programa por uma variável `NUM_REFLECTION_SAMPLES` que controla um laço de repetição onde a contribuição de cada um dos raios refletidos é adicionada ao resultado final. A variação do resultado devido a atribuição de diferentes valores a `NUM_REFLECTION_SAMPLES` pode ser visto na figura 6.6. Na figura a BSDF que controla as propriedades da esfera é *Torrance-Sparrow* com distribuição de *Blinn*.

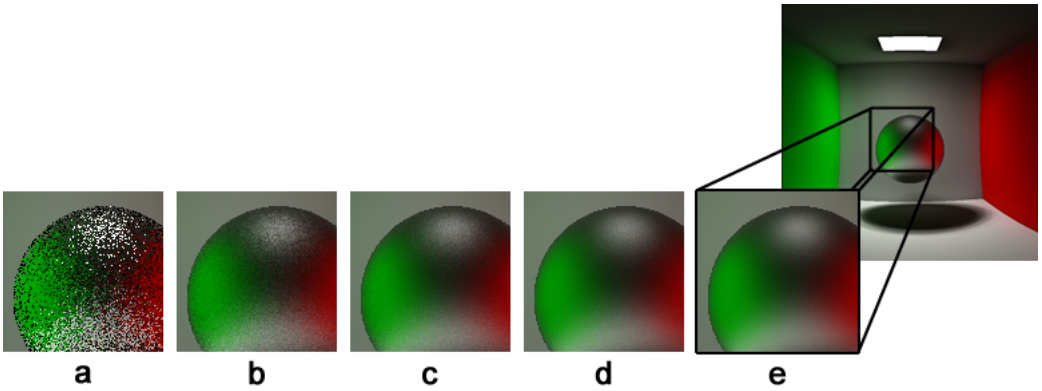


Figura 6.6: Detalhe da renderização de uma esfera com diferentes números de amostras para reflexão *Torrance-Sparow* utilizando a distribuição de *Blinn*. (a) 1 amostra; (b) 50 amostras; (c) 200 amostras; (d) 500 amostras e (e) 1000 amostras por reflexão.

O problema do ruído gerado pelos métodos de *Monte Carlo raytracing* abordado em diversas partes deste trabalho pode ser claramente percebido analisando os resultados gerados a partir das variações do número de raios refletidos na superfície especular. O aumento do valor de `NUM_REFLECTION_SAMPLES` elimina gradativamente o ruído gerado ao diminuir a variância inserida pelo método de *Monte Carlo*.

6.4 Photon Mapping

A estrutura de dados *photon map* foi baseada na implementação disponibilizada por Jensen [13], em C++.

6.4.1 Configurações do Photon Mapping

No algoritmo de *photon mapping*, é necessário atribuir valores a alguns parâmetros que influenciam tanto no passo de *photon tracing*, quanto no passo de renderização. O valor para cada um desses parâmetros são atribuídos em um arquivo texto de configuração, carregado no início da execução do programa. Um exemplo deste arquivo é exposto a seguir:

config.txt

```
1  pm_radius_search    = 5
2  pm_n_photons_search = 10000
3  pm_max_photons            = 1000000
4
5  photon_tracer_depth = 10
6
7  input_file = pm-scene-01.txt
8
9  render_type = global_illumination
10 filter_type = cone_filter
```

O primeiro parâmetro `PM_MAX_PHOTONS` determina o número de fótons a serem lançados no passo de *photon tracing*. Quando existe mais que uma fonte de luz, esse número é divido entre essas fontes, de forma proporcional a energia de cada uma delas, pois espera-se que fontes de luz com maior energia tenha maior influência no resultado final (ver figura 6.7). Os outros dois parâmetros `PM_NUM_PHOTONS_SEARCH` e `PM_RADIUS_SEARCH`, referentes ao cálculo da radiância no passo da renderização, são, respectivamente, o número máximo de fótons a serem buscados no *photon map* e o raio máximo dessa busca. Na Figura 6.8 uma comparação entre os resultados gerados com diferentes valores desses parâmetros pode ser visto.

Com o uso dos valores da Figura 6.8a, a radiância calculada nas superfície dos objetos contêm uma considerável variação devido ao número de fótons lançados ser relativamente baixo, provocando uma distribuição desigual da luz pela cena, e assim gerando efeitos visuais indesejados. Aumentando esses valores, como mostrado na Figura 6.8b, o resultado melhora consideravelmente, contando com uma boa representação da iluminação indireta. Porém, com uma análise minuciosa nota-se ainda pequenas variações ao longo da cena, principalmente perto dos cantos das paredes e no teto, que é predominantemente iluminado pela luz indireta. Com os valores da Figura 6.8c, nota-se que a iluminação é distribuída de forma uniforme pela cena, e a variação da radiância ao longo da cena é suave, inclusive no teto.

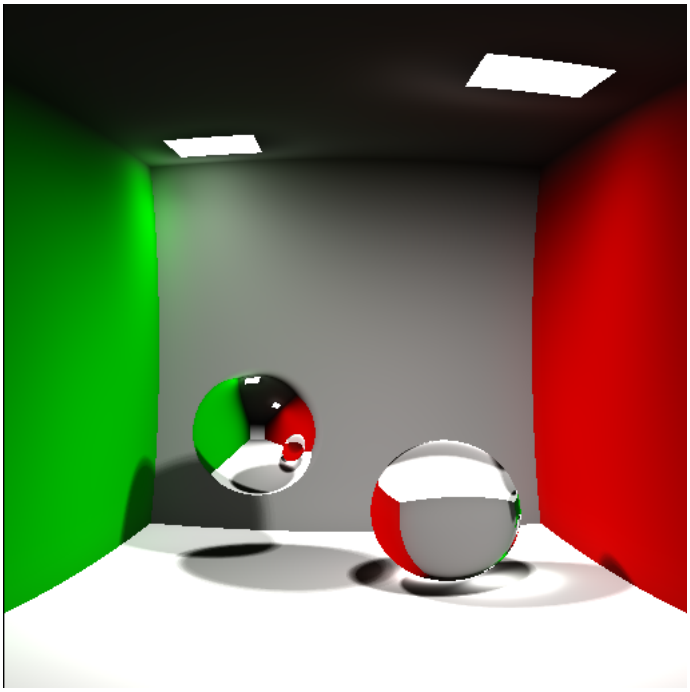


Figura 6.7: Renderização utilizando duas fontes de luz.

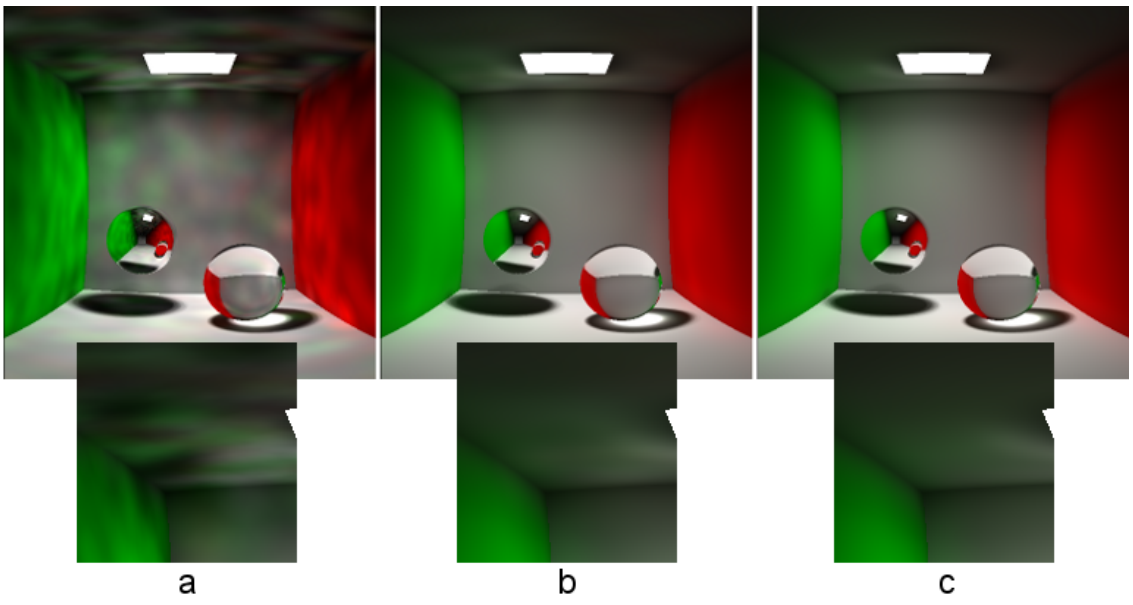


Figura 6.8: Resultado do algoritmo de *photon mapping* com os valores para `PM_NUM_PHOTONS`, `PM_NUM_PHOTONS_SEARCH` e `PM_RADIUS_SEARCH` respectivamente: a)10000, 100, 1; b)100000, 1000, 3; c)1000000, 10000, 5.

Além das partes implementadas mostradas neste capítulo, existem obviamente classes para efetuar os passos de *photon tracing*, *raytracing* entre outras operações necessárias para a geração dos resultados. Estas não são tratadas neste capítulo por serem comuns às várias implementações dos algoritmos com propósitos similares, e não conterem nenhuma especificidade a ser discutida.

No arquivo de configuração, existe também a opção de mudar o parâmetro `RENDER_TYPE` para `photon_map`, permitindo uma visualização rápida da distribuição dos fótons na cena, sem a etapa de renderização. Finalizando, o parâmetro `FILTER_TYPE` pode ser setado com os valores `NONE` ou `CONE_FILTER` determinando o uso ou não do filtro apresentado na subseção 4.3.1. Diversos outros filtros podem ser implementados, como o gaussiano por exemplo, sendo a inclusão desse parâmetro um facilitador para se determinar qual dos filtros deve ser usado ao se executar o programa.

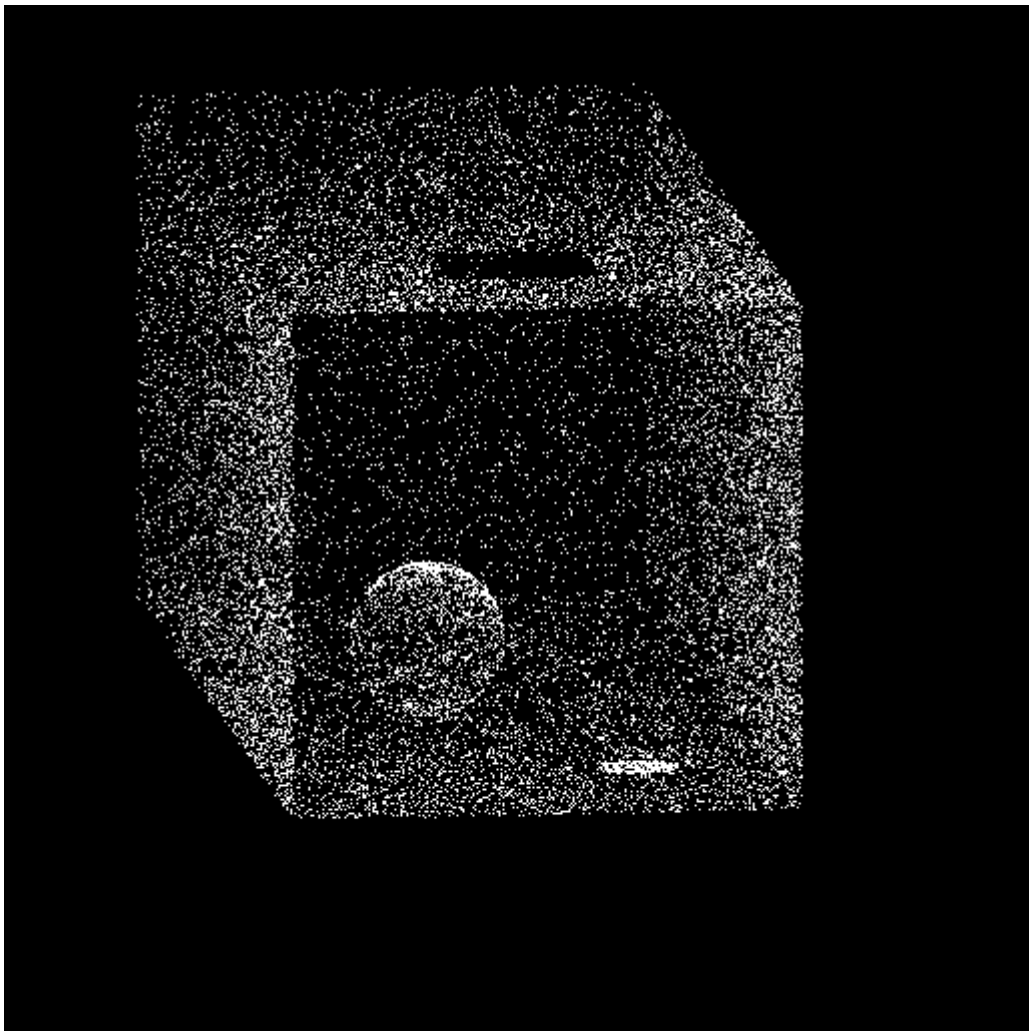


Figura 6.9: Renderização da representação simplificada do *photon mapping*

O arquivo de cena, por sua vez, descreve os objetos de uma forma intuitiva, atribuindo a cada um as propriedades do modelo de iluminação desejável. Um trecho do arquivo de cena utilizado para gerar as imagens da Figura 6.8 é exposto a seguir:

pm-scene-01.txt

```
1 #scene
2 ambientLight = 0.4 0.4 1
3 clearColor = 1 1 1
```

```
4  enviromentRefractionIndex = 1
5
6  #camera
7  observer_position = 0 0 -18
8  observer_up = 0 1 0
9  grid_position = 0 0 -17
10 xAxis = -1 0 0
11 yAxis = 0 1 0
12 zAxis = 0 0 1
13 width = 512
14 height = 512
15
16 #squarelight
17 position1 = -1.5 4.99 -1.5
18 position2 =  1.5 4.99 -1.5
19 position3 =  1.5 4.99  1.5
20 position4 = -1.5 4.99  1.5
21 normal = 0 -1 0
22 color = 1 1 1
23 power = 3500
24
25 #sphere
26 center = 2 -2 2
27 radius = 1.5
28 color = 1 1 1
29 reflectivity = 0.5
30 refractionIndex = 1
31 isotropy = 0
32 roughness = 30
33 BRDF = perfect_specular
34
35 #sphere
36 center = -2 -3.5 0
37 radius = 1.5
38 color = 1 1 1
39 reflectivity = 0.5
40 refractionIndex = 1.8
41 isotropy = 0
42 roughness = 30
43 BRDF = perfect_specular_t
44
45 #plane
46 position = -5 0 0
47 normal = 1 0 0
48 color = 1 0 0
49 reflectivity = 0.5
50 refractionIndex = 1
51 isotropy = 0
52 roughness = 30
53 BRDF = lambertian
54
55 #plane
56 position = 5 0 0
57 normal = -1 0 0
58 color = 0 1 0
59 reflectivity = 0.5
60 refractionIndex = 1
61 isotropy = 0
```

```
roughness = 30
BRDF = lambertian

#plane
position = 0 -5 0
normal = 0 1 0
color = 0.8 0.8 0.8
reflectivity = 0.5
refractionIndex = 1
isotropy = 0
roughness = 30
BRDF = lambertian

...
```

Apresentados os detalhes da implementação, o próximo capítulo mostra e discute os resultados gerados pelo programa apresentado.

Capítulo 7

Resultados

Utilizando o programa apresentado no Capítulo 6, foram renderizadas diversas cenas contendo objetos compostos por materiais caracterizados pelos modelos de iluminação apresentados no Capítulo 5. Associando a teoria apresentada com os respectivos resultados, torna-se possível uma compreensão de forma mais ampla e a verificação visual de como as especificidades de cada modelo refletem nas imagens renderizadas.

Apresentado na Seção 5.1, o modelo lambertiano descreve superfícies perfeitamente difusas, e como dito, apesar de não ser um modelo físico, apresenta bons resultados e é amplamente utilizado nos algoritmos de renderização. Na figura 7.1 foi renderizada uma cena onde todos os objetos são formados por superfícies lambertianas, refletindo igualmente a luz incidente em todas as direções.

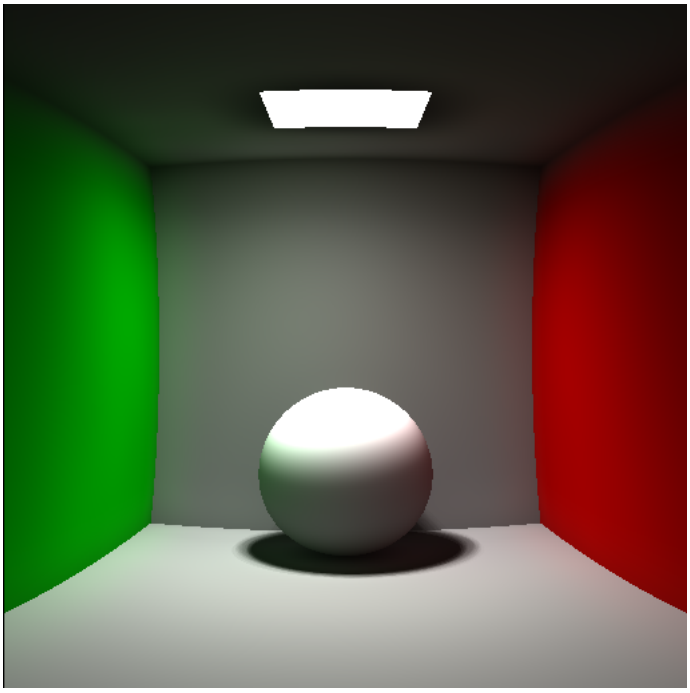


Figura 7.1: Reflexão Lambertiana

A Figura 7.2, mostra uma cena contendo uma esfera com superfície perfeitamente especular. Como explicado na Seção 5.2, a direção da luz incidente e da luz refletida formam um ângulo com a normal de mesmo valor, simulando as propriedades de um espelho perfeito. A aparência da esfera é determinado pela espelhamento da cena que a envolve, resultado esperado para uma superfície com essas características.

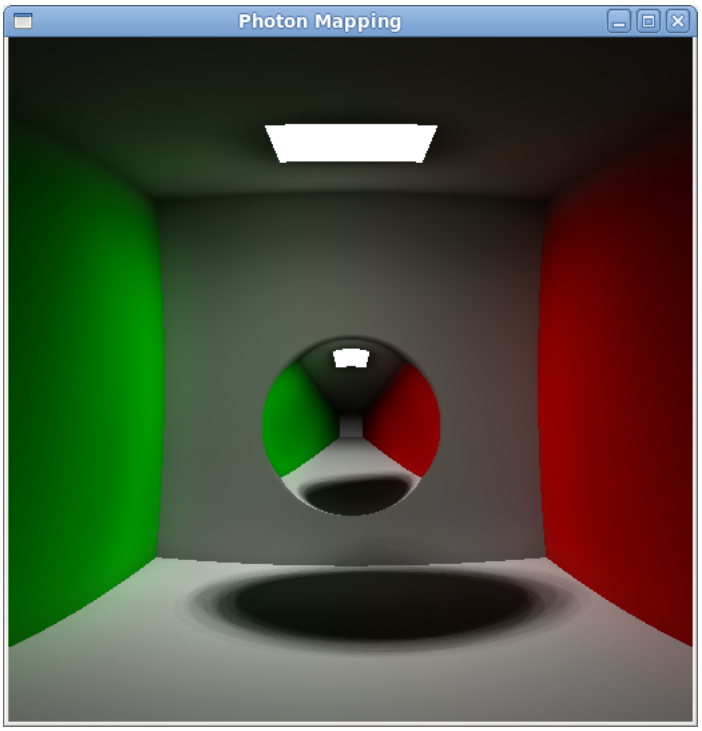
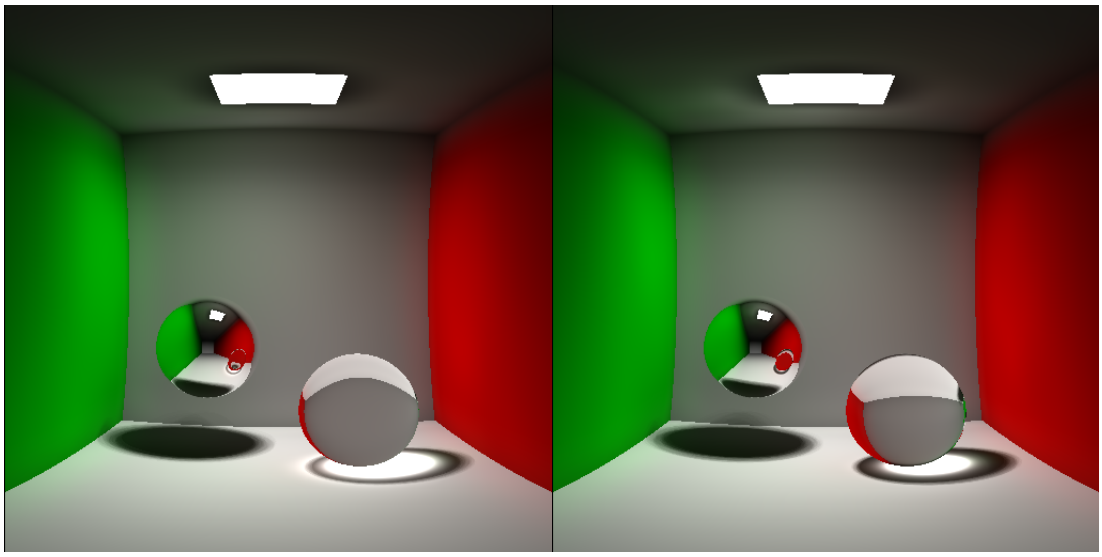


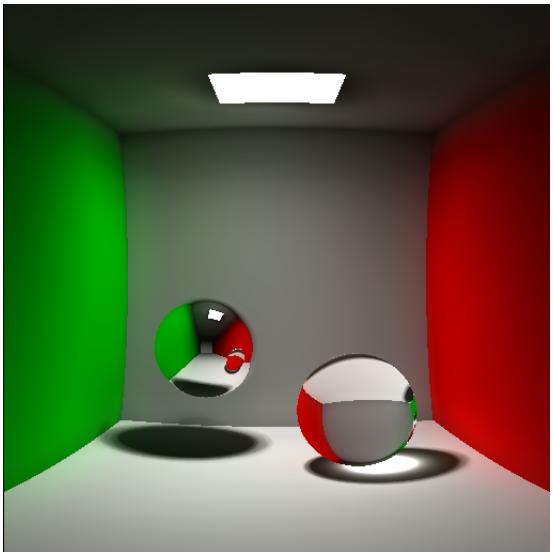
Figura 7.2: Reflexão Especular Perfeita

Na Figura 7.3, o ambiente contém um índice de refração $\eta = 1$ e a esfera no lado direito das cenas mostradas, contém em suas variações, diferentes índices de refração, gerando uma aparência singular para cada uma delas. Enquanto na Figura 7.3a ($\eta = 1.3$), a luz é transmitida de forma menos concentrada, formando uma cáustica que cobre uma área maior da superfície, na Figura 7.3c ($\eta = 1.9$) a luz transmitida se concentra em uma área menor da superfície. Como dito na Seção 5.2, a direção de transmissão da luz é determinada pela lei de Snell, então ao se aumentar o índice de refração do material, o seno do ângulo entre a direção de transmissão e a normal na superfície diminui, provocando então essa concentração da luz. Nota-se também, que esse comportamento, como esperado, não só influencia na aparência da cáustica, como também na da própria esfera, pela forma como a luz refletida nas superfícies sobrepostas é transmitida antes de chegar ao observador.

A Figura 7.4, mostra resultados obtidos utilizando o modelo de reflexão proposto por Oren-Nayar descrito na Seção 5.4, tanto na esfera, quanto nas paredes que a envolvem. Foram utilizados diferentes valores de rugosidade no material em cada uma das figuras, variando entre 10 e 50. Como pode ser notado, com o acréscimo da ru-



(a) $\eta = 1.3$ (b) $\eta = 1.6$

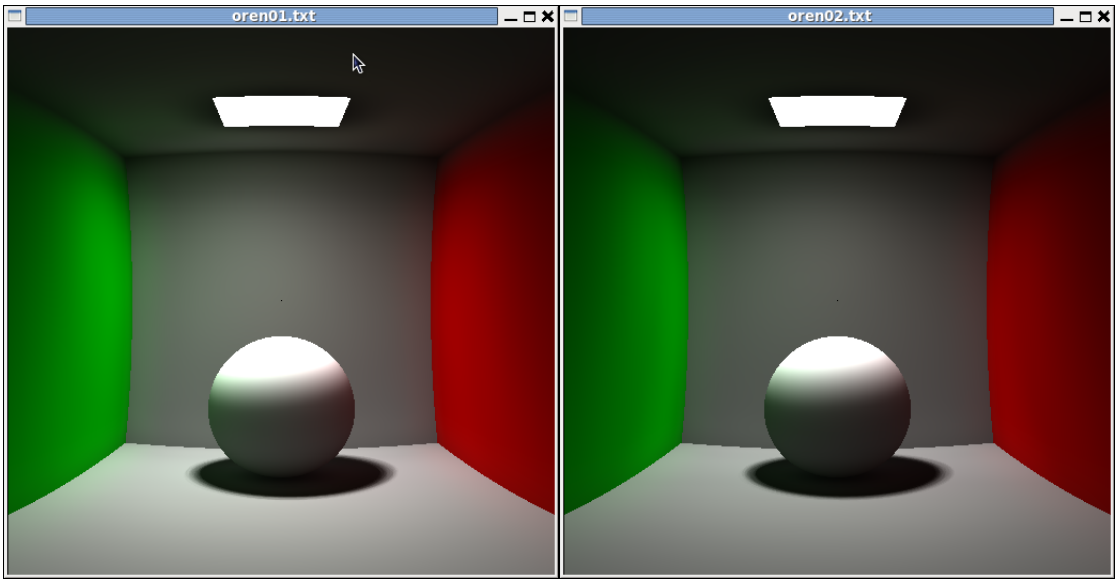


(c) $\eta = 1.9$

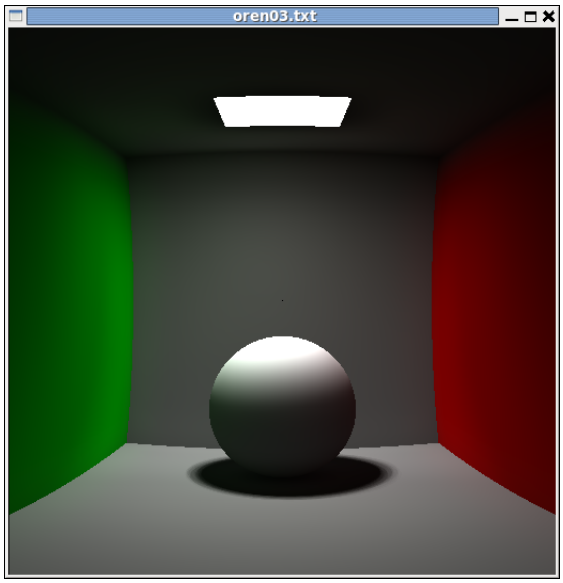
Figura 7.3: Transmissão especular, com índice de refração do ambiente igual a 1, e diferentes valores do índice de refração para a esfera direita.

gosidade nas superfícies a quantidade de luz refletida é menor devido aos fenômenos de sombreamento, oclusão e inter-reflexões entre as microfacetas.

Na figura 7.5, a interação entre a luz e a superfície da esfera tem seu comportamento definido pelo modelo de Torrance-Sparrow utilizando a função de distribuição de Blinn. Para cada uma das cenas mostradas foi utilizado um valor específico para o expoente e da função de distribuição de Blinn. Como explicado na subseção 5.5.1, a variação desse expoente, que define as normais das microfacetas, na prática, determina se a reflexão será mais *glossy*, ou mais próxima da reflexão especular perfeita (quanto menor o valor do expoente, mais rugosa a superfície e consequentemente, mais *glossy* será a reflexão). Esse comportamento pode ser claramente verificado na



(a) $\sigma = 10$ (b) $\sigma = 30$

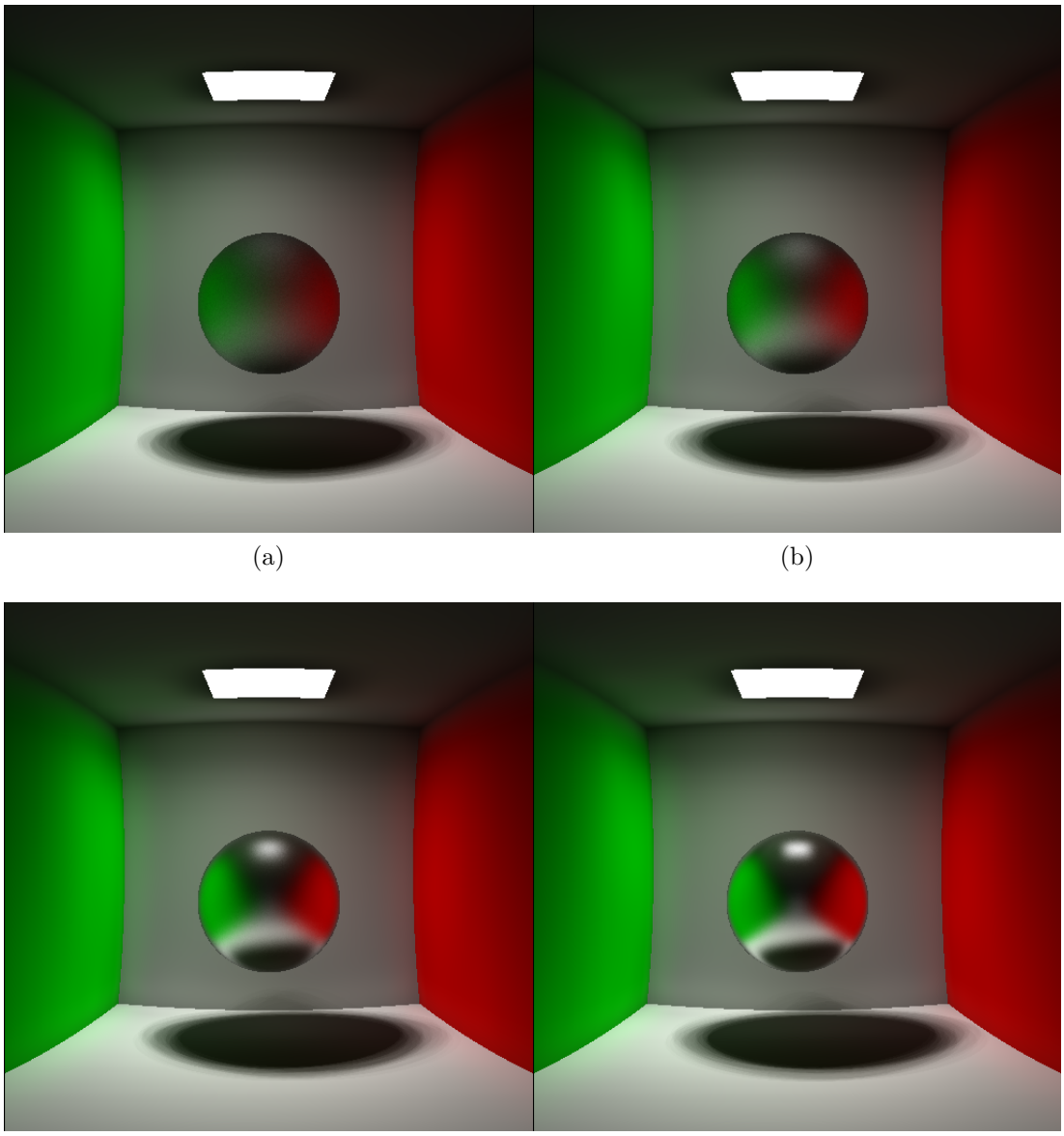


(c) $\sigma = 50$

Figura 7.4: Oren-Nayar

figura 7.5: enquanto na figura 7.5a, o valor de e é 5, dando um aspecto áspero para a superfície da esfera, na figura 7.5d, com e igual a 100, nota-se o visual correspondente a uma superfície lisa.

A figura 7.6 mostra a renderização de uma cena contendo uma esfera com propriedades definidas pelo modelo Torrance-Sparrow utilizando a função de distribuição anisotrópica, apresentada na subseção 5.5.2. A interação entre a luz e a superfície da esfera assume características diferentes a medida em que os valores dos termos e_x e e_y variam. Quando os termos tem valores iguais, o comportamento da distribuição é semelhante ao modelo de Blinn, como mostram as figuras 7.6a e 7.6d. Quando



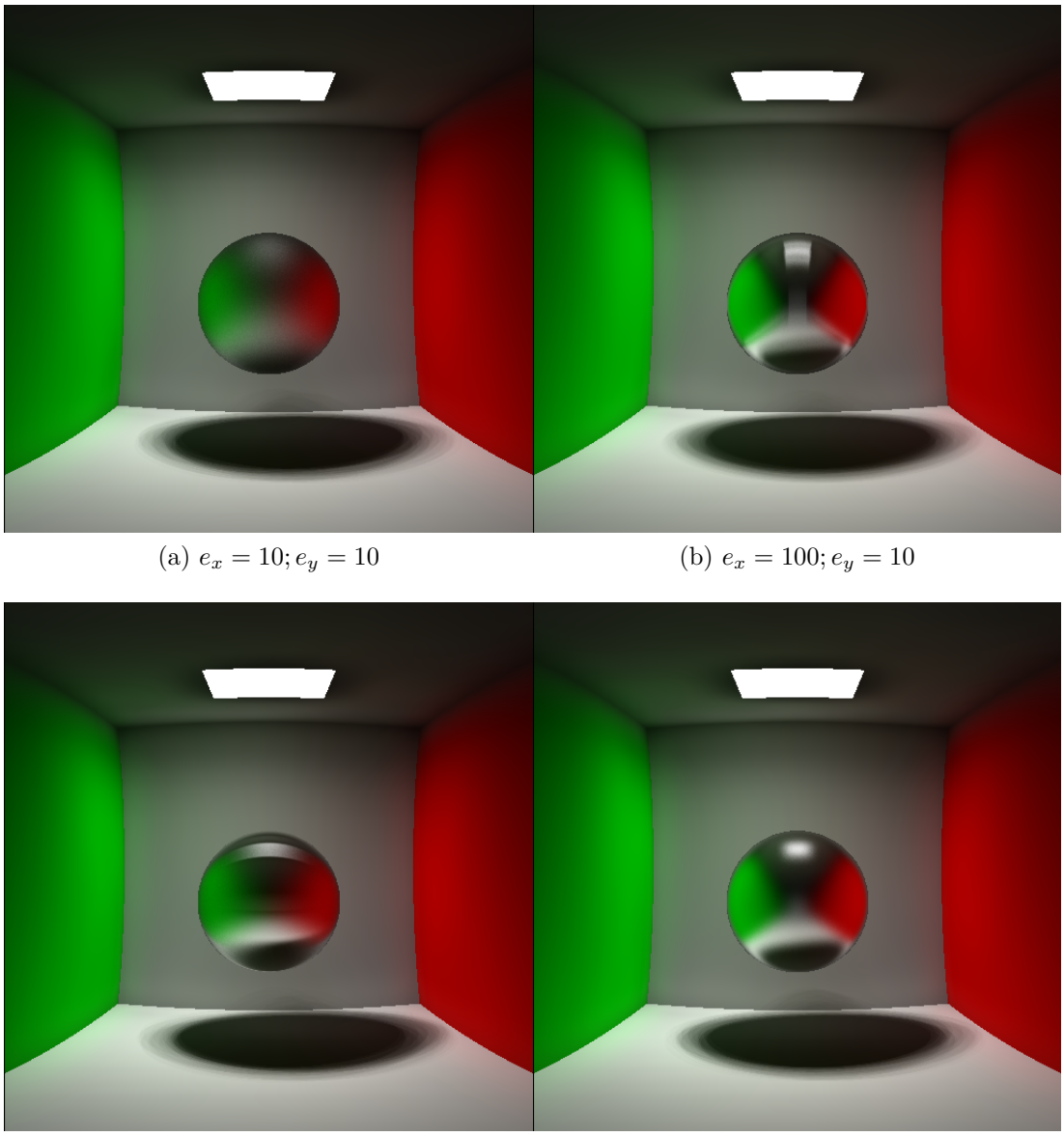
(a) (b)

(c) (d)

Figura 7.5: Torrance-Sparrow com distribuição Blinn

existe uma diferença entre os termos, a tendência é se ter a aparência de superfícies polidas na direção determinada pelo menor termo, como visto nas figuras 7.6b e 7.6c. Como nota-se também, quanto maior a diferença entre os termos, mais acentuado é este efeito.

Como dito na Seção 5.6, um dos méritos do modelo proposto por Schlick é a possibilidade de se simular diversos tipos de materiais, como rugosos, lisos, isotrópicos, anisotrópicos e diversas variações entre os citados. Na Figura 7.7 são mostrados os resultados do modelo de Schlick com o termo ψ referente a isotropia fixado com o valor 1, ou seja, isotrópico, e com o termo σ referente a rugosidade variando. Determinado esse cenário, o que se espera é um comportamento similar ao obtido com o modelo de Torrance-Sparrow utilizando a função de distribuição de Blinn,



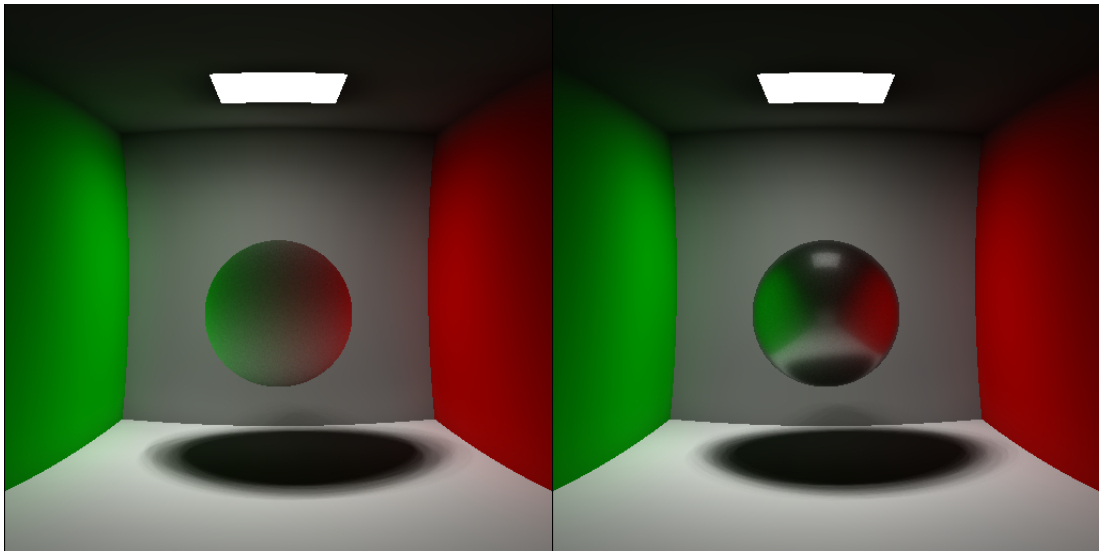
(a) $e_x = 10; e_y = 10$ (b) $e_x = 100; e_y = 10$

(c) $e_x = 10; e_y = 1000$ (d) $e_x = 100; e_y = 100$

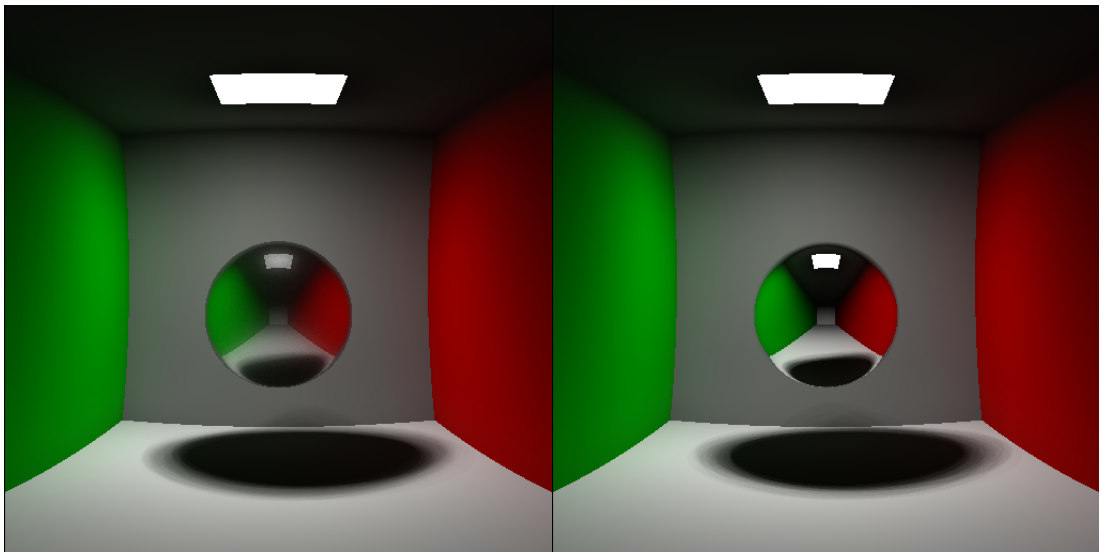
Figura 7.6: Torrance-Sparrow com distribuição Anisotrópica

pois como dito, este simula materiais isotrópicos com rugosidade. O que pode ser visualizado na Figura 7.7 é exatamente este comportamento: na Figura 7.7a, com o fator σ de rugosidade igual a 1, o aspecto da superfície da esfera é áspero, enquanto na figura 7.7b, tem o aspecto de uma superfície lisa. É importante ressaltar que apesar do comportamento similar esperado, não existem relações diretas entre as características dos materiais das esferas mostradas na Figura 7.5 e na Figura 7.7.

A Figura 7.8 mostra uma cena com diversos objetos (esfera especular, esfera transparente, e malha triangular com superfície lambertiana), onde podem ser visualizados alguns efeitos da iluminação global, como a iluminação indireta, *color bleeding* e cáusticas.



(a) $\psi = 1.0; \sigma = 1.0$ (b) $\psi = 1.0; \sigma = 0.5$



(c) $\psi = 1.0; \sigma = 0.2$ (d) $\psi = 1.0; \sigma = 0.05$

Figura 7.7: Schlick

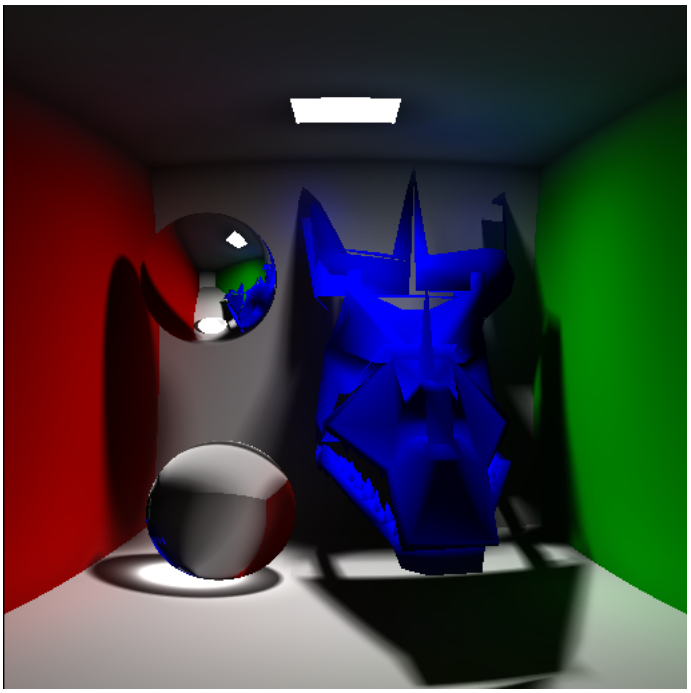


Figura 7.8: Renderização contendo esferas especular e transparente e malha de triângulos lambertiana.

Capítulo 8

Conclusões e Trabalhos Futuros

Neste trabalho foi apresentada uma implementação e um estudo do método de simulação da iluminação global proposto por Jensen [19], o *photon mapping*. Foi realizado um estudo comparativo entre o uso de diversos tipos de BRDF neste contexto, dado que este algoritmo tem como uma das suas principais vantagens lidar com qualquer tipo de função de reflectância. Pelas renderizações expostas no capítulo 7, nota-se que de fato é possível simular diversos tipos de materiais, obtendo também outros efeitos esperados pela iluminação global, como *color bleeding* e cáusticas, por exemplo.

O fator determinante na complexidade do algoritmo não é o número de fótons lançados na cena, mas sim a renderização utilizando o algoritmo de *ray-tracing*. A razão principal é o custo da busca dos n fótons mais próximos ao ponto de interseção de um raio. Outro fator fundamental é o número de raios amostrados a partir de reflexões para os modelos de iluminação mais complexos como, por exemplo, o de Schlick. São necessárias algumas centenas de amostras para obter um resultado minimamente suave e plausível. As imagens geradas para estes modelos foram realizadas com 1000 amostras por reflexão, ao invés de apenas uma como no caso de modelos lambertianos ou especulares perfeitos; esta modificação eleva o tempo de renderização de alguns minutos para algumas horas. De fato, o algoritmo para gerar o mapa de fótons requer apenas alguns segundos, mesmo para cenas com milhões de fótons.

A implementação de um código modular permitiu a fácil extensão e modificação das funções de iluminação e distribuição. Este grande peso dedicado à parte prática durante esta dissertação foi um fator fundamental para a compreensão das minúcias e problemas referentes à simulação da luz em uma cena sintética. Apenas assim foi possível atingir um grau de clareza quanto aos conceitos fundamentais, o que provavelmente não seria possível somente pela análise teórica do tema. Isto é especialmente notável quanto ao entendimento dos parâmetros nos diversos modelos de iluminação. Alguns conceitos e peculiaridades dos métodos computacionais para

aproximar a integral de iluminação também ficam mais evidentes a partir dos esforços necessários para implementá-los.

Um dos grandes desafios dos algoritmos de iluminação global, além da simulação realística da luz, é a possibilidade de ser ser calculada em tempo real, característica importante em aplicações como jogos. Com esse objetivo, nos últimos anos foram desenvolvidos extensões do *photon mapping* utilizando programação em placas gráficas. Um dos primeiros trabalhos nesta direção foi o de Purcell et al. [29], e entre as diversas propostas conseguintes, estão as de realizar o *photon mapping* em tempo real de Fabianowski e Dingliana [30] e o *photon mapping* em espaço de imagem de McGuire e Luebke [31]. Um ponto importante desses trabalhos é como implementar estruturas de dados eficientes para o mapa de fótons em GPU [32, 33]. Um trabalho futuro natural é a pesquisa, desenvolvimento e implementação deste algoritmo utilizando placas gráficas com o propósito de atingir renderização em tempo real, ou pelo menos, em tempos consideravelmente inferiores.

De fato, este é um ponto da tese que não foi dado a merecida atenção, já que não foi priorizado nenhum tipo de eficiência em termos de tempo. Apesar das malhas triangulares terem sido inseridas como prova de conceito, é um caso onde este problema fica especialmente evidente por não ter sido utilizada nenhuma estrutura espacial para acelerar o teste de interseção do raio com o objeto. Com um empenho adicional na otimização do código, poderão ser testadas também cenas mais complexas, visto que elevar muito o número de objetos na implementação atual pode tornar os tempos de renderização proibitivos.

Entre demais trabalhos futuros, está a extensão do método de *photon mapping* para que o mesmo possa lidar com influência do meio (*participating media*), como descrito por Jensen [13]. Isso possibilitaria a renderização realística de ambientes com fumaça, poeira, além de materiais translúcidos, como plantas e mármore. Além disso, um trabalho já em andamento, é a geração a mistura de diferentes BRDFs para um mesmo objeto, tornando-os mais interessantes e podendo alcançar renderizações que aproximem novos tipos de materiais reais. A Figura 8.1 ilustra um primeiro exemplo neste sentido.

Por fim, espera-se que o trabalho comparativo exposto nesta dissertação possa ser útil a futuros interessados em ingressar na área de iluminação global, especialmente através da disponibilização do sistema implementado, que pode beneficiar aqueles que desejem ingressar nesta fascinante área da computação gráfica.

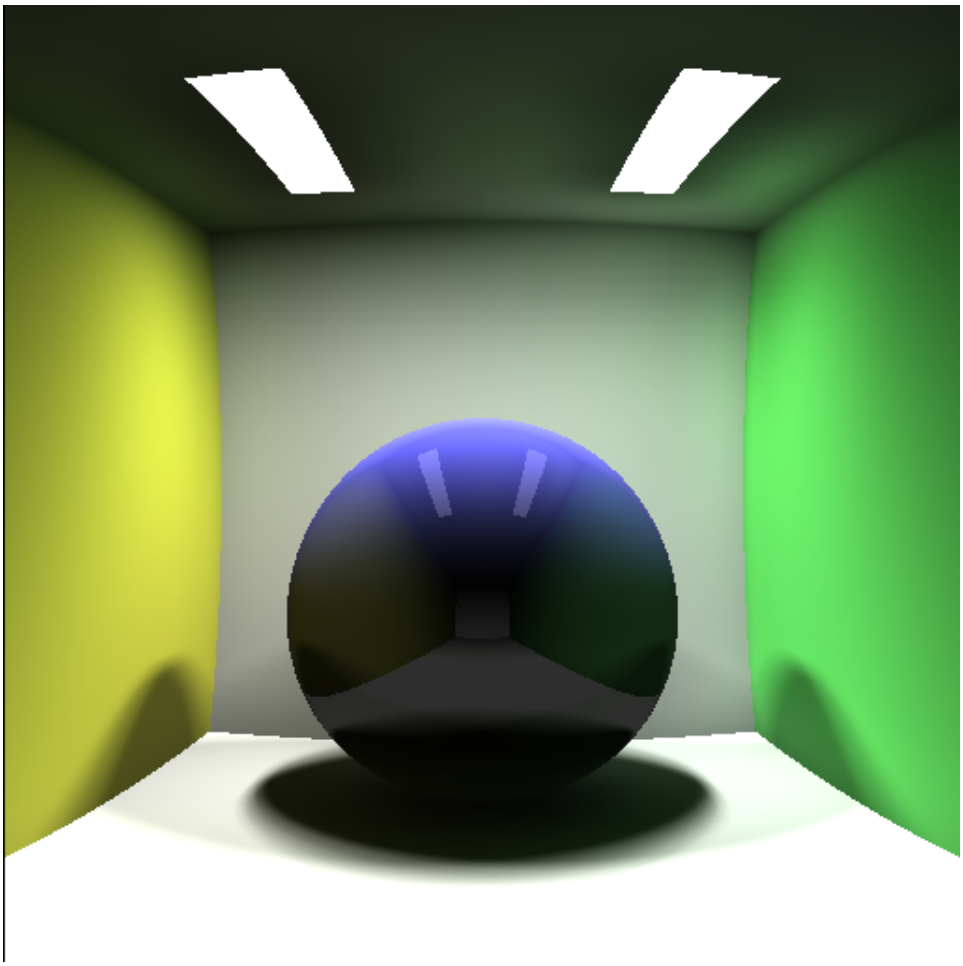


Figura 8.1: Material da esfera representado pela mistura de uma BRDF Lambertiana e uma especular.

Referências Bibliográficas

[1] PHARR, M., HUMPHREYS, G. *Physically Based Rendering: From Theory to Implementation*. San Francisco, CA, USA, Morgan Kaufmann Publishers Inc., 2004. ISBN: 012553180X.

[2] WHITTED, T. "An improved illumination model for shaded display", *Commun. ACM*, v. 23, n. 6, pp. 343–349, 1980. ISSN: 0001-0782. doi: http://doi.acm.org/10.1145/358876.358882.

[3] COOK, R. L., PORTER, T., CARPENTER, L. "Distributed ray tracing", *SIGGRAPH Comput. Graph.*, v. 18, n. 3, pp. 137–145, 1984. ISSN: 0097-8930. doi: http://doi.acm.org/10.1145/964965.808590.

[4] GORAL, C. M., TORRANCE, K. E., GREENBERG, D. P., et al. "Modeling the interaction of light between diffuse surfaces", *SIGGRAPH Comput. Graph.*, v. 18, n. 3, pp. 213–222, 1984. ISSN: 0097-8930. doi: http://doi.acm.org/10.1145/964965.808601.

[5] WALLACE, J. R., COHEN, M. F., GREENBERG, D. P. "A two-pass solution to the rendering equation: A synthesis of ray tracing and radiosity methods", *SIGGRAPH Comput. Graph.*, v. 21, n. 4, pp. 311–320, 1987. ISSN: 0097-8930. doi: http://doi.acm.org/10.1145/37402.37438.

[6] SILLION, F., PUECH, C. "A general two-pass method integrating specular and diffuse reflection". In: *SIGGRAPH '89: Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, pp. 335–344, New York, NY, USA, 1989. ACM. ISBN: 0-89791-312-4. doi: http://doi.acm.org/10.1145/74333.74368.

[7] CHEN, S. E., RUSHMEIER, H. E., MILLER, G., et al. "A progressive multi-pass method for global illumination". In: *SIGGRAPH '91: Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, pp. 165–174, New York, NY, USA, 1991. ACM. ISBN: 0-89791-436-8. doi: http://doi.acm.org/10.1145/122718.122737.

[8] TORRANCE, K. E., SPARROW, E. M., BIRKEBAK, R. C. "Polarization, Directional Distribution, and Off-Specular Peak Phenomena in Light Reflected from Roughened Surfaces", *Journal of the Optical Society of America (1917-1983)*, v. 56, pp. 916–+, 1966.

[9] BLINN, J. F. "Models of light reflection for computer synthesized pictures", *SIGGRAPH Comput. Graph.*, v. 11, n. 2, pp. 192–198, 1977. ISSN: 0097-8930. doi: http://doi.acm.org/10.1145/965141.563893.

[10] COOK, R. L., TORRANCE, K. E. "A Reflectance Model for Computer Graphics", *ACM Trans. Graph.*, v. 1, n. 1, pp. 7–24, 1982. ISSN: 0730-0301. doi: http://doi.acm.org/10.1145/357290.357293.

[11] KAJIYA, J. T. "The rendering equation", *SIGGRAPH Comput. Graph.*, v. 20, n. 4, pp. 143–150, 1986. ISSN: 0097-8930. doi: http://doi.acm.org/10.1145/15886.15902.

[12] JENSEN, H. W., ARVO, J., DUTRE, P., et al. "Monte Carlo Ray Tracing". . Siggraph 2003 Course 44, July 2003.

[13] JENSEN, H. W. *Realistic image synthesis using photon mapping*. Natick, MA, USA, A. K. Peters, Ltd., 2001. ISBN: 1-56881-147-0.

[14] TALBOT, J. F., CLINE, D., EGBERT, P. K. "Importance resampling for global illumination". In: Bala, K., Dutré, P. (Eds.), *Rendering Techniques 2005 Eurographics Symposium on Rendering*, pp. 139–146, Aire-la-Ville, Switzerland, 2005. Eurographics Association. ISBN: 1727-3463.

[15] ZANG, A. R. *Esquema híbrido para amostragem de mapas de iluminação em renderizações foto-realistas*. Tese de Mestrado, IMPA - Instituto Nacional de Matemática Pura e Aplicada, July 2009.

[16] LAFORTUNE, E. P., WILLEMS, Y. D. "Bi-Directional Path Tracing". In: *Proceedings of Third International Conference on Computational Graphics and Visualization Techniques (Compugraphics '93*, pp. 145–153, 1993.

[17] VEACH, E., GUIBAS, L. J. "Metropolis light transport". In: *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pp. 65–76, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co. ISBN: 0-89791-896-7. doi: http://doi.acm.org/10.1145/258734.258775.

[18] JENSEN, H. W. J., CHRISTENSEN, N. J. "Photon maps in bidirectional Monte Carlo ray tracing of complex objects", *Computers & Graphics*,

v. 19, n. 2, pp. 215 – 224, 1995. ISSN: 0097-8493. doi: http://dx.doi.org/10.1016/0097-8493(94)00145-O.

[19] JENSEN, H. W. "Global illumination using photon maps". In: *Proceedings of the eurographics workshop on Rendering techniques '96*, pp. 21–30, London, UK, 1996. Springer-Verlag. ISBN: 3-211-82883-4.

[20] ARVO, J., KIRK, D. "Particle transport and image synthesis". In: *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, pp. 63–66, New York, NY, USA, 1990. ACM. ISBN: 0-89791-344-2. doi: http://doi.acm.org/10.1145/97879.97886.

[21] BENTLEY, J. L. "Multidimensional Binary Search Trees in Database Applications", *IEEE Trans. Softw. Eng.*, v. 5, n. 4, pp. 333–340, 1979. ISSN: 0098-5589. doi: http://dx.doi.org/10.1109/TSE.1979.234200.

[22] LARSON, G. W. "Real Pixels", *Graphics Gems II*, pp. 80–83, 1991.

[23] PREPARATA, F. P., SHAMOS, M. I. *Computational geometry: an introduction*. New York, NY, USA, Springer-Verlag New York, Inc., 1985. ISBN: 0-387-96131-3.

[24] OREN, M., NAYAR, S. K. "Generalization of Lambert's reflectance model". In: *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pp. 239–246, New York, NY, USA, 1994. ACM. ISBN: 0-89791-667-0. doi: http://doi.acm.org/10.1145/192161.192213.

[25] ASHIKHMIN, M., SHIRLEY, P. "An Anisotropic Phong BRDF Model", *Journal of Graphics Tools*, v. 5, pp. 25–32, 2000.

[26] SCHLICK, C. "A Customizable Reflectance Model for Everyday Rendering". In: *In Fourth Eurographics Workshop on Rendering*, pp. 73–83, 1993.

[27] OPENGL. `http://www.opengl.org`, 2010.

[28] TOOLKIT, G. T. O. U. `http://http://www.opengl.org/resources/libraries/glut/`, 2010.

[29] PURCELL, T. J., DONNER, C., CAMMARANO, M., et al. "Photon Mapping on Programmable Graphics Hardware". In: *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware*, pp. 41–50. Eurographics Association, 2003. ISBN: 1-58113-739-7.

[30] FABIANOWSKI, B., DINGLIANA, J. “Interactive Global Photon Mapping”, *Computer Graphics Forum*, v. 28, n. 4, pp. 1151–1159, 2009.

[31] MCGUIRE, M., LUEBKE, D. “Hardware-accelerated global illumination by image space photon mapping”. In: *HPG ’09: Proceedings of the Conference on High Performance Graphics 2009*, pp. 77–89, New York, NY, USA, 2009. ACM. ISBN: 978-1-60558-603-8. doi: http://doi.acm.org/10.1145/1572769.1572783.

[32] CZUCZOR, S., SZIRMAY-KALOS, L., SZÉCSI, L., et al. “Photon map gathering on the GPU”. . Eurographics Short Papers, 2005.

[33] LEFOHN, A., KNISS, J. M., OWENS, J. D. “Implementing Efficient Parallel Data Structures on GPUs”. In: Pharr, M. (Ed.), *GPU Gems 2*, Addison-Wesley, pp. 521–545, Reading, Massachusetts, March 2005. ISBN: 0321335597.