Capítulo 10 - *Rendering*

1. Introdução

A Computação Gráfica trata da síntese de imagens no computador, o que envolve as etapas de modelagem e de *rendering*. Existem vários tipos de processos de *rendering*, sendo que vamos nos concentrar em técnicas para gerar imagens matriciais de cenas tridimensionais. Para sintetizar imagens é necessário partir de uma descrição da cena em termos da definição dos objetos que a compõem (geometria dos objetos da cena, informações sobre os materiais de que são feitos esses objetos, como cor, a textura, etc.); das condições de iluminação ambiente; e do ponto de observação. Podemos interpretar o processo de *rendering* como o de converter modelos gráficos em uma imagem. Malhas poligonais e superfícies paramétricas são exemplos de modelos gráficos que podem ser usados para compor cenas.

O Processo Físico

A Figura 10.1 ilustra, de forma simplificada, o que acontece quando olhamos para um objeto, ou para uma cena. Os raios de luz são emitidos, por uma fonte de luz, em todas as direções. Alguns destes raios atingem o objeto que estamos observando. O objeto absorve parte da luz que o atinge, e o restante é refletido por sua superfície. Parte desta luz refletida pode atingir os nossos olhos: quando isso ocorre, "vemos" o objeto.

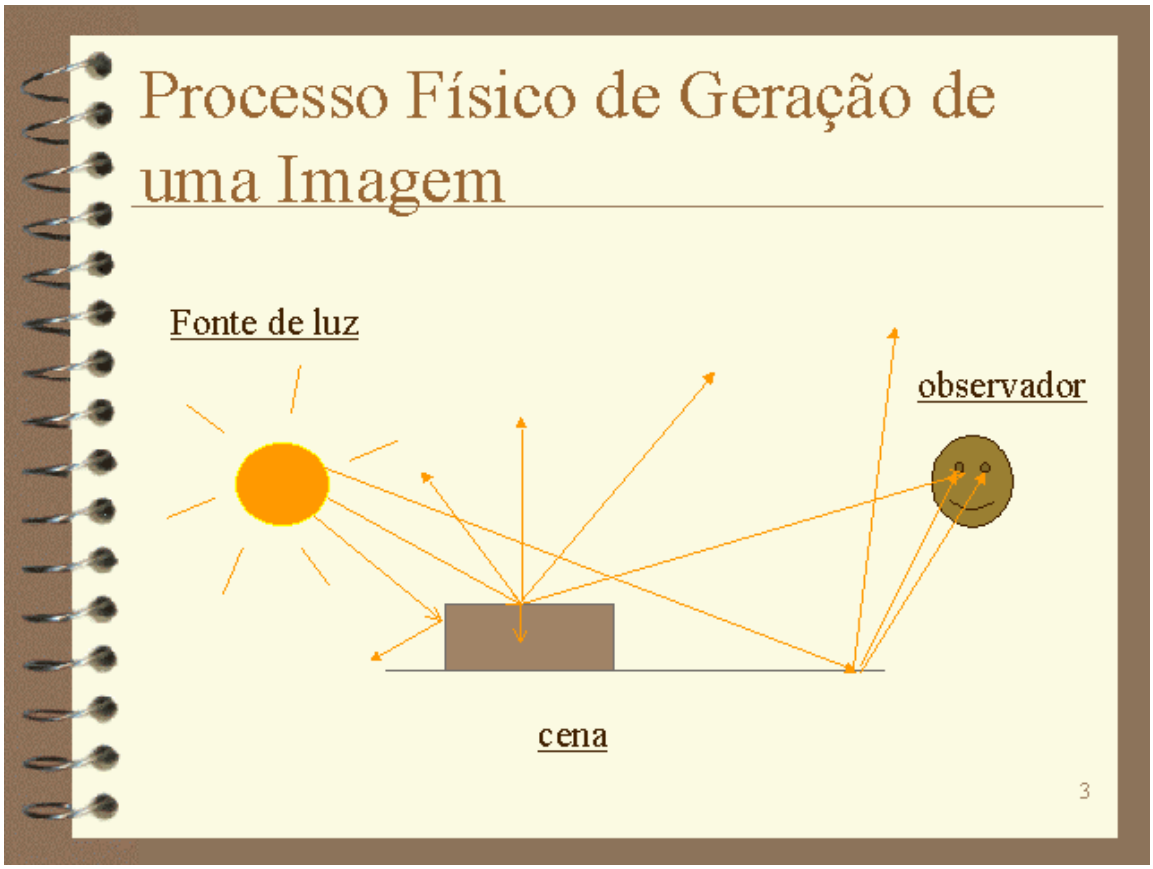

Figura 1. Processo Físico de geração de uma imagem.

Uma técnica comum usada em computação gráfica, e bastante eficiente, é a emissão de raios, ou *ray-tracing*. O *ray-tracing* simula a interação dos raios de luz com o ambiente, seguindo o caminho percorrido por cada raio. Observe que a fonte de luz emite uma enorme quantidade de raios, sendo que poucos destes atingem de fato o nosso olho. Simular este processo computacionalmente, seguindo o traçado de um grande número de raios, dos quais uma pequena parte apenas contribui de fato para o resultado, teria um custo proibitivo. Por isso, seguimos o caminho do raio ao contrário: a partir do ponto em que ele atinge o olho do observador, voltamos para determinar quais objetos ele atingiu desde que saiu da fonte de luz. Quando o raio atinge o objeto em um certo ponto, podemos determinar se o ponto está sendo iluminado pela fonte, traçando o raio a partir do ponto de intersecção até a fonte de luz. Se o raio atinge a fonte, esse ponto está sendo diretamente iluminado. A idéia é lançar pelos menos um raio para cada pixel que compõe a imagem, e traçá-lo de volta (bem como todos os raios "gerados" por ele na interação com os objetos da cena) até que ele

intercepte uma fonte de luz. O *ray tracing* gera imagens foto-realistas (ou quase), mas para isso incorre em um alto custo computacional, e é lento (geralmente é implementado em software). Existem outras abordagens, em particular o algoritmo *scanline,* que produz resultados mais limitados, mas tem um custo computacional menor.

Métodos *Image-Order* x *Object-Order*

O *ray-tracing é um* método do tipo *image-order*, que funciona tentando determinar o que acontece para cada raio de luz, um de cada vez (ou seja, determina a imagem pixel a pixel: para pixel, descobre com que cor ele deve ser 'pintado'). Um processo do tipo *object-order* funciona renderizando os objetos que compõem a cena, um de cada vez. No exemplo acima, um processo deste tipo tentaria renderizar o chão, e depois o cubo (de forma análoga a que um pintor pintaria um quadro). Em geral, este tipo de processo pode ser implementado em hardware (existe hardware dedicado para renderizar certos tipos de primitivas, como triângulos, por exemplo), garantindo sua rapidez.

Surface Rendering* x *Volume Rendering

Observe que ao renderizar os objetos, consideramos a interação de suas superfícies com os raios de luz em termos dos raios que são refletidos e transmitidos pela superfície. Ou seja, renderizamos as superfícies apenas, sendo que o interior do objeto não é considerado explicitamente. O interior está delimitado pela fronteira, ou superfície, do objeto, mas não é visível (como ocorre com a maioria dos objetos do nosso dia a dia). Esse tipo de *rendering* é denominado *surface rendering*, sendo que vamos nos concentrar nele. Entretanto, esta estratégia não é adequada para renderizar objetos como nuvens, água e neblina, que são translúcidos e deixam a luz passar por eles, "esparramando" os raios de luz em todas as direções. Nesse caso, usa-se outra estratégia denominada *volume rendering*, que considera também a interação da luz com o interior dos objetos.

2. Cor

O *rendering* de imagens coloridas requer algum tipo de representação de cor. A forma de armazenar e representar cores no computador pode ser vista, de certa forma, como uma simplificação da forma que o olho humano usa para reconhecer as cores. O espectro

luminoso visível para os seres humanos contém comprimentos de onda entre 400 e 700 nm. A luz que penetra os nossos olhos consiste de diferentes intensidades destes comprimentos de onda, sendo que o gráfico de intensidades do comprimento de onda no intervalo 400-700nm define a cor da luz. Na verdade, o olho humano ignora a maior parte desta informação: existem 3 tipos de receptores de cores no nosso olho, sendo que cada tipo responde a um subconjunto do intervalo de intensidades entre 400-700nm. Simplificadamente, um tipo de receptor absorve intensidades em uma faixa na região do espectro associada à cor azul, outro na faixa do verde, e outro na faixa do vermelho.

Dois sistemas bastante usados na representação computacional de cores são o RGB (*Red, Green, Blue*) e o HSV (*Hue, Saturation, Value*). O sistema RGB representa as cores com base nas suas intensidades de vermelho, verde e azul. Este sistema pode ser visto como um espaço tridimensional em que os 3 eixos correspondem às intensidades de Vermelho, Verde e Azul. O sistema HSV representa as cores com base no Matiz, Saturação e Brilho. O componente de brilho (ou intensidade) representa a quantidade de "luz" presente na cor (0 para preto, ou nenhum brilho, 1 para cor brilhante). O matiz define o comprimento de onda dominante da cor (ilustrado geralmente através de um círculo no qual o ângulo de rotação representa o matiz). Pode ser descrito como um valor entre 0 e 1, sendo que 0 corresponde a 0 graus no círculo, e 1 corresponde a 360 graus. A saturação indica a quantidade de matiz acrescentada à cor (0 nenhum matiz, cor fica branca; 1 o máximo, tem-se uma cor primária). Por exemplo, se V = 1, H = 0.66 (região cujo comprimento de onda dominante é o azul), e S = 1, tem-se um azul primário (puro) brilhante. Se S = 0.5, temo-se um azul celeste intenso. Se S = 0, tem-se o branco.

3 Fontes de Luz

A definição das fontes de luz é essencial para o *rendering*: sem elas, qualquer imagem da cena ficaria toda preta! É a interação entre a luz emitida pelas fontes e os objetos que compõem a cena que define o que vemos da cena. Uma fonte de luz pode ser considerada simplesmente como mais um objeto na especificação da cena, sendo que se distingue pelo fato de emitir luz diretamente. Na especificação de fontes de luz deve-se definir as seguintes grandezas:

• **Geometria** - formato físico da fonte;

• **Intensidade** - função que associa a cada ponto do espaço a intensidade luminosa da fonte de luz nesse ponto;

• **Distribuição espectral** - contribuição da fonte em cada comprimento de onda (cor) do espectro visível.

Quanto ao aspecto geométrico pode-se classificar as fontes de luz em 3 tipos:

• **Fontes direcionais** - a fonte de luz é considerada no infinito, e pode ser determinada por um vetor unitário que define a sua direção. Em geral são utilizadas para aproximar fontes de luz pontuais a uma distância infinita, como por exemplo, o sol.

• **Fontes pontuais** - são definidas por um vetor posição *(x,y,z)* no espaço de cena, e um vetor unitário que define a sua direção. Esta tem dimensões desprezíveis se comparadas à dimensão dos objetos da cena.

• **Fontes de área** - possuem uma superfície não pontual emissiva, e um sistema de coordenadas locais associado que é utilizado para especificar a sua posição no espaço e a direção de iluminação.

Vamos considerar o caso mais simples: uma fonte pontual, que é uma simplificação da luz incandescente que usamos normalmente em casa. O posicionamento no infinito é interessante porque implica que todos os raios de luz que atingem a cena são paralelos entre si - o que não é o caso se a fonte for pontual e local à cena. Normalmente, assume-se que a intensidade da luz emitida por esta fonte permanece constante a medida em que percorre o espaço, ou então que decresce segundo um fator de atenuação linear com a distância percorrida. Isso não corresponde ao modelo real do processo físico, que afirma que a luz é atenuada segundo um fator inversamente proporcional ao quadrado da distância percorrida.

4 Propriedades das Superfícies - Um Modelo de Iluminação

Ao atravessarem o espaço, alguns dos raios de luz interceptam os objetos da cena. Quando isso ocorre, os raios interagem com a superfície conforme as propriedades do objeto, de forma a produzir uma cor. Vamos apresentar uma visão geral do fenômeno físico da propagação da energia luminosa e de sua interação com a superfície de um objeto. Devido à complexidade e alto custo computacional dos modelos reais de iluminação, o modelo físico apresentado é uma aproximação simplificada para o fenômeno real que conserva um bom grau de realismo nas imagens geradas a partir de sua aplicação.

Quando uma energia luminosa incide sobre a superfície de um objeto, ela pode ser **absorvida**, **refletida** ou **transmitida**. Uma parte da energia luminosa incidente na superfície é absorvida e convertida em calor. A parte restante é refletida ou transmitida, sendo que a reflexão ou transmissão da luz é que torna um objeto visível. Se toda a luz incidente é absorvida, o objeto é invisível e é chamado de corpo negro.

A característica da luz refletida pela superfície de um objeto depende da composição, direção e geometria da fonte de luz, da orientação da superfície e das propriedades da superfície do objeto. A luz refletida por um objeto é também caracterizada por ser refletida **difusamente** ou **especularmente**, dependendo das propriedades da superfície.

A reflexão especular é o resultado de uma reflexão perfeita, cujo efeito é análogo a uma bola elástica pulando sobre uma superfície lisa. O ângulo segundo o qual a bola deixa a superfície é determinado pelo ângulo segundo o qual a bola a atinge (ângulo de incidência). A reflexão difusa é como uma bola elástica pulando sobre uma superfície completamente rugosa, onde não se pode determinar *a priori* a direção que a bola irá tomar. Ao contrário da reflexão especular, a luz refletida difusamente é emitida pela superfície com igual intensidade em todas as direções, sendo de menor intensidade que a luz incidente. Há dois tipos extremos de superfícies que se distinguem quando se considera esse fenômeno:

- **Refletoras Idealmente Especulares**: aquelas que atuam como perfeitos espelhos (Ex.: metais polidos, água parada).

- **Refletoras Idealmente Difusas**: aquelas que são opacas ou foscas (Ex.: cortiça ou gesso);

Entretanto, grande parte dos objetos possui superfícies com características intermediárias entre estes dois extremos. A seguir, é apresentado o desenvolvimento de um modelo de iluminação simples, que considera inicialmente a reflexão de luz direta por uma superfície idealmente difusa.

A *lei dos co-senos de Lambert* governa a reflexão de uma luz proveniente da fonte de luz por uma superfície **idealmente difusa**. Esta lei determina que a intensidade da luz refletida por uma superfície idealmente difusa é proporcional ao co-seno do ângulo entre a direção da luz incidente e a normal à superfície. Especificamente,

$$I = I_1 K_d \cos\theta \qquad 0 \le \theta \le \frac{\pi}{2}$$

onde I é a intensidade da luz refletida, I_l é a intensidade da luz emitida pela fonte e incidente à superfície, K_d $(0 \le K_d \le 1)$ é a constante de difusão da superfície e θ é o ângulo entre a direção da luz incidente **L** e a normal à superfície **N** (figura 1). O coeficiente K_d depende do material e do comprimento de onda da luz incidente. Entretanto, para modelos de iluminação simples, pode ser assumido como sendo constante.

Como esse modelo assume que apenas a fonte de luz é emissora de energia luminosa, os objetos que não recebem luz diretamente da fonte aparecem escuros ou pretos. Entretanto, em uma cena real cada objeto também recebe luz refletida indiretamente pelos outros objetos da cena, o que caracteriza uma fonte de luz distribuída. A fim de simplificar os cálculos associados a essa iluminação indireta, considera-se uma componente de luz constante vinda do meio ambiente. Esta luz ambiente pode ser acrescentada à equação de Lambert como um termo constante de difusão que caracteriza o componente de luz ambiente. O modelo de iluminação fica, então:

$$I = I_a K_a + I_l K_d \cos\theta \qquad 0 \le \theta \le \frac{\pi}{2}$$

onde *Ia* é a intensidade da luz ambiente incidente e K_a $(0 \leq K_a \leq 1)$ é a constante de reflexão difusa da luz ambiente. Observe que, se o modelo acima for usado para determinar a intensidade de luz refletida por dois objetos iluminados pela mesma fonte de luz, mas a distâncias diferentes dessa fonte, o resultado seria a mesma intensidade de luz refletida para ambos os objetos. Como vimos, isto é uma simplificação, pois é sabido que a intensidade da luz decresce de forma inversamente proporcional ao quadrado da distância do objeto em relação à fonte. Alguns modelos usam uma constante de atenuação linear (ver Rogers). Se a superfície é colorida, o modelo de iluminação é aplicado individualmente para calcular as intensidades de cada uma das três primárias (vermelho, verde e azul). A seguir, o modelo acima é estendido para incluir a emissão de luz refletida **especularmente** pelos objetos.

A equação que governa a reflexão especular é a *equação de Fresnel*, que afirma que a reflexão especular da luz é direcional. Para uma superfície idealmente especular, o ângulo de reflexão será igual ao ângulo de incidência e toda energia refletida será na direção do raio refletido (**R**). Isto implica que o vetor de visão **S** (Figura 2), é coincidente com o vetor de reflexão **R**; isto é, o ângulo α é zero. Porém, para superfícies em geral a luz é refletida em todas as direções de um modo não uniforme e aleatório, com intensidade dependendo do ângulo do raio incidente. Este tipo de reflexão não uniforme é também denominada reflexão não coerente. Para superfícies polidas (próximas às idealmente especulares), a distribuição espacial dos raios de luz refletidos especularmente é limitada ou focalizada, criando o efeito conhecido como *highlight*, enquanto que para superfícies rugosas a luz refletida é espalhada.

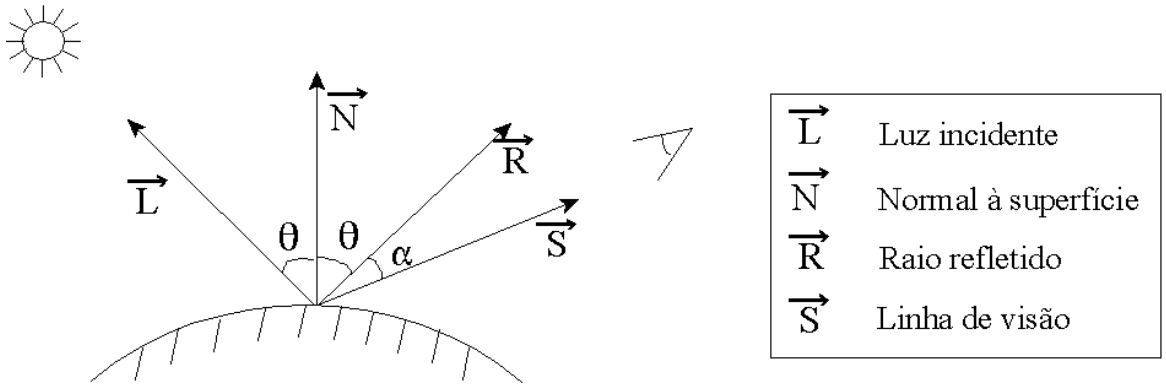

Figura 2. Reflexão Especular.

O modelo empírico de *Bui-Tuong Phong* é freqüentemente usado em modelos simples de iluminação. Especificamente,

$$I_s = I_l W(i,\lambda)\cos^n \alpha$$

onde $W(i,\lambda)$ é a curva de reflexão que dá a razão entre a luz refletida especularmente e a luz incidente em função do ângulo de incidência i e do comprimento de onda λ, e n é a aproximação da distribuição espacial da luz refletida especularmente. Valores grandes de n caracterizam distribuições espaciais de metais e outras superfícies especulares. Valores pequenos de n caracterizam superfícies não metálicas e opacas.

A reflexão especular é **direcional**, isto é, depende do ângulo com que a luz incidente atinge a superfície. Para alguns materiais não metálicos a reflexão pode ser pequena, em torno de 4%, enquanto que para materiais metálicos pode exceder 80%. Combinando os resultados obtidos anteriormente, tem-se um modelo de iluminação dado pela equação:

$$I = I_a K_a + I_l (K_d \cos\theta + K_s \cos^n \alpha) \quad 0 \le \theta \le \frac{\pi}{2}$$

onde Ks é uma constante freqüentemente utilizada para representar $W(i,\lambda)$. Se múltiplas fontes de luz estiverem presentes, os efeitos são linearmente adicionados e o modelo de iluminação torna-se

$$I = I_a K_a + \sum_{j=1}^{m} I_{l_j} (K_d \cos\theta_j + K_s \cos^n \alpha_j) \qquad 0 \le \theta \le \frac{\pi}{2}$$

onde m é o número de fontes de luz. Utilizando o produto escalar de dois vetores, pode-se escrever:

$$\cos\theta = \frac{\mathbf{N} \cdot \mathbf{L}}{|\mathbf{N}||\mathbf{L}|} = \hat{\mathbf{N}} \cdot \hat{\mathbf{L}}$$

onde $\hat{\mathbf{N}}, \hat{\mathbf{L}}$ são as unidades vetoriais da normal à superfície e da direção da fonte de luz, respectivamente.

Similarmente,

$$\cos\alpha = \frac{\mathbf{R} \cdot \mathbf{S}}{|\mathbf{R}||\mathbf{S}|} = \hat{\mathbf{R}} \cdot \hat{\mathbf{S}}$$

onde $\hat{\mathbf{R}}, \hat{\mathbf{S}}$ são unidades vetoriais para a luz refletida e a direção da linha de visão. Então, o modelo de iluminação para uma única fonte de luz é dado por:

$$I = I_a K_a + I_l (K_d (\hat{\mathbf{N}} \cdot \hat{\mathbf{L}}) + K_s (\hat{\mathbf{R}} \cdot \hat{\mathbf{S}})^n)$$

Em computação gráfica este modelo é freqüentemente chamado de função de iluminação. A função de iluminação é aplicada individualmente a cada uma das três cores primárias em uma imagem colorida. Isso é uma simplificação (em geral, a reflexão varia para cada componente espectral), mas é simples de implementar e produz resultados aceitáveis.

5 Tonalização

O *rendering* da cena pode ser feito aplicando-se a equação de iluminação acima a cada ponto (pixel) da superfície de cada objeto que compõem a cena. Na verdade, isso é feito a cada ponto que aparece de fato na imagem da cena a ser gerada, ou a cada ponto visível. Isso implica em que o processo deve ser precedido de algum procedimento para determinar os pontos não visíveis, um seja, um procedimento **de remoção de superfícies ocultas**. No caso de uma cena modelada por conjuntos de polígonos, o algoritmo *scanline* para preenchimento de polígonos pode ser aplicado, e o modelo de iluminação é usado para calcular a intensidade (cor) a ser associada a cada pixel. Observe que esse processo de *rendering* é baseado em **modelo local de iluminação** - ou seja, fora a luz ambiente, modelada como uma constante no modelo, apenas os raios de luz que incidem diretamente

sobre cada ponto da superfície são considerados no cálculo da cor. A interação dos raios refletidos ou transmitidos com os demais objetos da cena é ignorada. O *ray-tracing*, por outro lado, considera estas interações, e por isso diz-se que é baseado em um modelo global de iluminação.

As técnicas de tonalização (*shading*) utilizam algum modelo de iluminação, por exemplo, o que foi visto acima, para determinar as cores associadas aos pixels de cada polígono que compõe uma cena. As técnicas mais comumente utilizadas são a **Tonalização Constante**, ou *Flat Shading*, a **Tonalização por Gouraud**, ou *Gouraud Shading*, e a **Tonalização por Phong**, ou *Phong Shading*.

Na tonalização constante, são considerados apenas os termos da iluminação ambiente e da reflexão difusa, com uma única fonte de luz posicionada no infinito e na direção de visão do observador, de modo a evitar o surgimento de sombras. O processo é um dos mais simples e mais eficientes, porém, não considera o fato de que, em geral, a representação poliedral de um modelo representa uma aproximação linear por partes do modelo real. Ele assume que a representação poligonal é o modelo real em estudo. Desse modo, cada polígono representa uma parte do modelo onde o vetor normal é constante, e, portanto a intensidade de luz calculada usando a equação local de iluminação também é constante no polígono. Daí o termo *Flat Shading*, devido à aparência facetada da imagem resultante.

As técnicas desenvolvidas por Gouraud e Phong permitem a obtenção de uma aparência mais suave, sendo, portanto mais utilizadas em superfícies que representam uma aproximação linear da superfície realmente desejada. Ambas consideram as componentes de iluminação ambiente, difusa e especular. A técnica de Gouraud consiste em aplicar o modelo de iluminação para calcular as intensidades nos vértices do polígono, e interpolar os valores obtidos para obter a iluminação ao longo de cada aresta e nos pontos interiores. A técnica de Phong calcula as normais nos vértices, e interpola as normais para calcular o modelo de iluminação em cada pixel do polígono. Ela apresenta os melhores resultados para superfícies especulares, mas também é a de custo computacional mais alto.

6 Transparência

O modelo de iluminação apresentado anteriormente considera apenas objetos e superfícies opacas. Entretanto, nem todos os materiais utilizados para confeccionar objetos são opacos: alguns, como o vidro e a água, transmitem luz. Quando um raio de luz passa de um meio para outro, normalmente ocorre o fenômeno de refração. A intensidade com que o raio de luz é inclinado na refração é governada pela lei de Snell,

$$\eta_1 \sin\theta = \eta_2 \sin\theta'$$

onde η_1 e η_2 são os índices de refração no primeiro e segundo meios, respectivamente, θ é o ângulo de incidência e θ ' é o ângulo de refração. Nenhum material transmite toda a luz incidente. Uma parte da luz é refletida.

Por analogia com reflexão especular e difusa, a luz pode ser transmitida especularmente ou difusamente. Materiais transparentes transmitem a luz especularmente, enquanto que materiais que transmitem a luz difusamente têm aparência fosca ou translúcida.

As implementações mais simples do efeito de transparência ignoram a refração e seus efeitos (ver Rogers). Efeitos simples de transparência podem ser diretamente incorporados ao processo de tonalização. A idéia é primeiro marcar todas as superfícies transparentes. Quando uma superfície visível é transparente, uma combinação linear das duas superfícies mais próximas é armazenada no *frame buffer* para ser mostrada. A intensidade é então,

$$I = tI_1 + (1-t)I_2 \quad 0 \le t \le 1$$

sendo I_1 a superfície visível, I_2 a superfície imediatamente atrás da superfície visível, e t o fator de transparência para I_1. Se I_2 também é transparente, o algoritmo é aplicado recursivamente até encontrar uma superfície opaca ou o fundo da cena. Essa aproximação linear não é adequada para superfícies curvas, ou objetos complexos que espalham a luz, como nuvens, etc.

7. Visão Geral do Algoritmo *Scanline*

Nesse algoritmo o cálculo da iluminação é integrado a uma abordagem de rasterização (i.e., conversão da descrição vetorial, dada por polígonos, em uma descrição matricial). A imagem é gerada a partir da geometria projetada (descrita no espaço de tela), pixel a pixel: Incorpora 3 sub-processos, executados de forma integrada:

(a) Rasterização: processo de determinar em quais pixels da tela um polígono é projetado. Na verdade, resolve o problema para cada pixel (em ordem de linha de varredura), determinando quais superfícies de objetos são projetadas nesse pixel.

(b) Determinação das superfícies visíveis: para todos os pontos que são projetados em um pixel é calculada a sua profundidade, para determinar qual deles é visível no pixel (determinação da superfície visível).

(c) Tonalização: aplica um modelo de iluminação no ponto visível para determinar a cor do pixel.

(a) Rasterização

É feita na ordem da imagem: processa a cena projetada no espaço de tela na ordem das linhas de varredura. Opera sobre quaisquer tipos de polígonos (côncavos, convexos, auto-interceptantes, com buracos, ...). Envolve 3 passos:

(a.1) Obter as intersecções da linha de varredura com todas as arestas de polígonos que a interceptam e montar a estrutura de dados. inteiro).

(a.2) Ordenar as intersecções em ordem crescente de coordenada x.

(a.3) Determinar os pixels interiores a polígonos (para estes, serão executados os passos (b) e (c) acima). Usa a regra da paridade ímpar (pixels são interiores se paridade é ímpar)

Algoritmo explora coerência de arestas (muitas linhas de varredura consecutivas interceptam as mesmas arestas). Mantém uma tabela de arestas ativas (AET): lista das arestas interceptadas pela linha de varredura corrente, em ordem crescente de coordenada x do ponto de intersecção. Operação: atualiza a AET (Tabela de Arestas Ativas, ou *Active Edge Table*) para cada linha de varredura. Para

a linha y+1: arestas que estão na AET, mas não são interceptadas pela linha y+1 são removidas; novas arestas interceptadas por y+1 são inseridas, novas coordenadas x de intersecção são calculadas, o conteúdo da linha corrente é rasterizado.

(b) Determinação de superfícies visíveis: *depth-buffer/z-buffer* é mais ou menos padrão em estações gráficas.

- Requer duas áreas de memória: o *frame-buffer* para armazenar as intensidades de cada pixel, o *z-buffer* para armazenar a profundidade do ponto visível nesse pixel.
- Inicialmente, o *z-buffer* é inicializado com profundidade *zmin*, e o *frame buffer* é inicializado com cor de fundo.
- Cada superfície listada na tabela de faces é rasterizada em uma ordem arbitrária (sabe-se em quais pixels essa superfície é projetada), e é processada por ordem de linha de varredura.

A medida em que processa as linhas de varredura, calcula a profundidade z de cada ponto que é projetado no pixel P(x,y) (usando a equação do plano que contém a face poligonal na qual está o ponto). Essa profundidade é comparada com a profundidade atualmente armazenada em *zbuffer*(x,y). Se é maior, essa nova profundidade é armazenada, e a intensidade da superfície (i.e., sua cor, calculada pelo modelo de iluminação) no ponto é armazenada em *frame-buffer*(x,y)

(c) Usa o modelo de iluminação de Phong (em geral), e algum método de tonalização.

Bibliografia

C.N.Lenz Cesar, Um Módulo Eficiente para a Visualização de Sólidos B-Rep (Diss. de Mestrado), ICMSC-USP, 1995.

D.F. Rogers, Procedural Elements for Computer Graphics

Foley et al., Computer Graphics, Principles and Practice

Schroeder et al., The Visualization Toolkit

Agma Traina e Maria Cristina Ferreira de Oliveira