1
Introdução

O efeito visual de sombras em uma cena 3D é muito enriquecedor. Sem ele, quando a imagem é projetada na tela bidimensional do computador para visualização, perdem-se informações importantes sobre a localização exata dos objetos no mundo virtual, como a distância entre eles e o solo ou entre dois objetos que aparecem um por trás do outro. A Figura 1 mostra como as sombras ajudam a esclarecer essas relações espaciais entre as entidades da cena. A imagem (a) poderia ser interpretada de diversas maneiras, sendo impossível dizer, por exemplo, se as esferas estão tocando o plano. Em (b) e (c), as posições relativas dos objetos são evidenciadas pelas sombras.

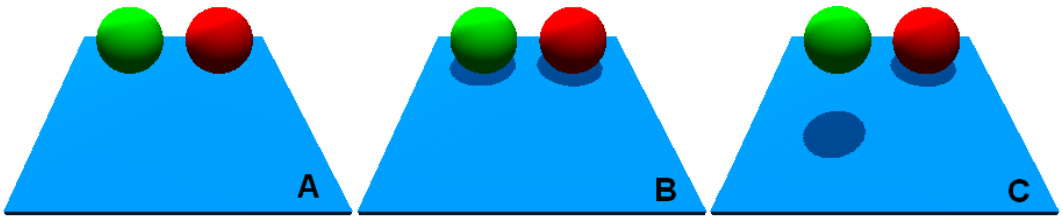

Figura 1 – Importância das sombras na localização espacial de objetos

Sombras são essenciais em visualizações realistas e podem compor a beleza estética de uma imagem artística. Em jogos, um ambiente sombreado pode tornar um local mais ameaçador ou misterioso, enquanto silhuetas de personagens fornecem dicas valiosas sobre suas posições. Em aplicações técnicas, como a visualização de modelos CAD (*Computer-Aided Design*), as sombras podem auxiliar na compreensão do modelo em estudo, esclarecendo as posições e dimensões dos objetos.

Infelizmente, sombras constituem um problema de iluminação global (*global illumination*), pois a sombra de cada objeto em relação a cada fonte de luz pode se projetar sobre quaisquer outros objetos da cena. Além disso, o processo de rasterização de triângulos do hardware gráfico atual não considera as relações espaciais entre múltiplas entidades nos cálculos de iluminação. Assim, a

determinação de sombras globais é um problema não-trivial, que já originou diferentes algoritmos e que ainda motiva muitas pesquisas em busca de qualidade visual e eficiência computacional.

1.1. Algoritmos de geração de sombras

Diversos algoritmos já foram propostos para lidar com a geração de sombras em cenas 3D. Citamos brevemente aqui alguns deles.

Algoritmos de iluminação global, como radiosidade (*radiosity*) [1] e rastreamento de raios (*ray tracing*) [2], e são os que produzem resultados mais fisicamente realistas. No entanto, radiosidade requer um longo tempo de pré-processamento, e ambos são incapazes de produzir imagens em tempo real para cenas complexas e dinâmicas.

Quando se deseja obter sombras de um conjunto conhecido de objetos em apenas uma ou em um pequeno número de superfícies, podem-se utilizar algoritmos de projeção direta. Como um exemplo, o seguinte processo permite a obtenção de sombras em superfícies curvas [3]: primeiro, desenha-se a cena do ponto de vista da fonte de luz em uma textura, os oclusores de preto e cada receptor de branco em um canal de cor. Depois, ao se renderizar a cena final, a textura projetada segundo a direção da luz indica os pontos dos receptores que devem estar em sombra.

Considerando-se o caso geral de uma cena complexa e dinâmica, com quase todos os objetos gerando sombras uns sobre os outros e sobre si próprios, dois tipos de algoritmos se destacam por poderem ser implementados em tempo real: Volumes de Sombra (*Shadow Volumes*) [4] e Mapeamento de Sombras (*Shadow Mapping*) [5]. Ambos são amplamente utilizados na indústria de jogos e outras aplicações de visualização.

A técnica de volumes de sombra trabalha no espaço de cada objeto oclusor, gerando uma representação poligonal de toda a região semi-infinita do espaço que está em sombra com relação a cada fonte de luz (Figura 2). Cada pixel da imagem final pode então ser testado, resultando em sombras precisas, sem serrilhamento (*aliasing*) e muito bem-definidas (*hard-shadows*). É importante perceber que este método não trata objetos não-poligonais, como aqueles cuja geometria é alterada

pelo teste de alfa ou por mapas de deslocamento (*displacement mapping*). Além disso, em cenas muito complexas, seu custo em processamento geométrico e em rasterização (*fill-rate*) pode se tornar bastante elevado.

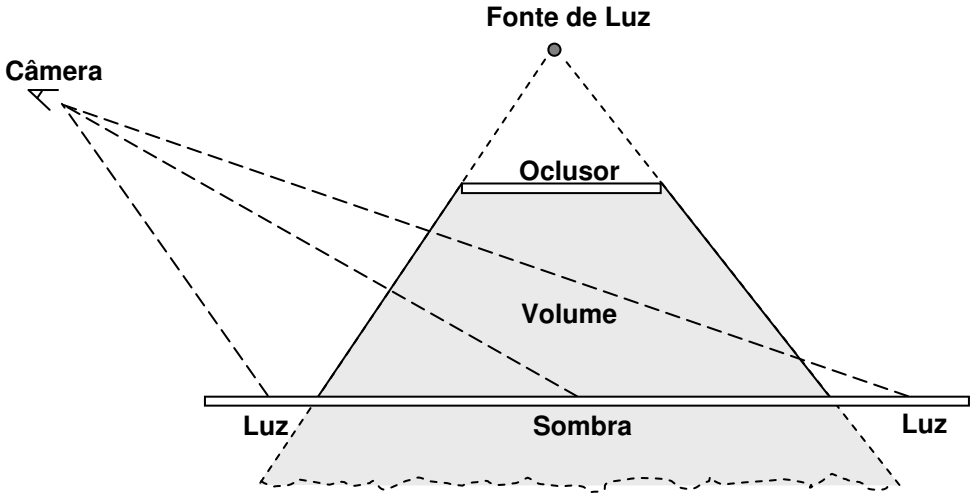

Figura 2 – Algoritmo de volumes de sombra

A técnica de mapeamento de sombras, adotada neste estudo, é atualmente a mais utilizada para a geração de sombras em superfícies arbitrárias. Trata-se de um método simples, fácil de implementar, com bom suporte em hardware e adequado a qualquer tipo de primitiva que possa marcar o *z-buffer*. O algoritmo, ilustrado na Figura 3, é composto pelas duas fases a seguir:

(A) Fase de atualização (ponto de vista da fonte de luz):
1 – Para cada fonte de luz:
1.1 – Renderizar a cena marcando apenas o *z-buffer*.
1.2 – Armazenar o *z-buffer* em uma textura de profundidade (*depth texture*), que contém então as distâncias da fonte até os oclusores mais próximos "vistos" por ela.

(B) Fase de renderização (ponto de vista da câmera):
2 – Desenhar a cena com iluminação ambiente e marcando o *z-buffer*.
3 – Para cada fonte de luz:
3.1 – Renderizar a cena com a iluminação da fonte somente nos pixels que passarem no teste de sombra (*shadow test*), acumulando cores com as passadas 2 e

3.1 anteriores. Para o teste de sombra, transforma-se cada pixel para o espaço da fonte de luz, projetado sobre o texel do mapa de sombras a ser acessado. Se a distância do pixel até a fonte for maior que o valor armazenado na textura, ele está em sombra. Caso contrário, está iluminado.

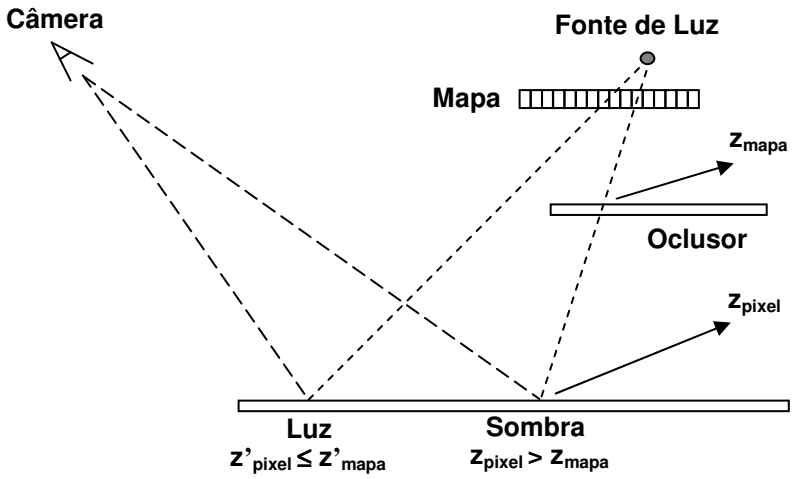

Figura 3 – Algoritmo de mapeamento de sombras

O mapeamento de sombras tem três problemas principais ausentes nos volumes de sombra: aparência serrilhada das bordas das sombras (*aliasing*), chamuscamento (*surface acne, self-shadow*), e dificuldade de renderizar a cena do ponto de vista de uma fonte de luz onidirecional para a geração de sombras em todas as direções.

1.2. Particularidades de modelos CAD

Quando consideramos a geração de sombras aplicada a modelos CAD complexos, precisamos ter em mente algumas de suas características particulares: primeiro, esse tipo de modelo geralmente contém muitos objetos estreitos, como cabos e barras de ferro, e objetos com silhuetas complexas, como treliças e grades. Dessa forma, precisamos tentar garantir que o mapa de sombras tenha resolução suficiente para capturar essas entidades, ou as sombras podem aparecer não só serrilhadas, mas também confusas e falhas. Além disso, esse tipo de cena muitas vezes possui um elevado grau de complexidade em z (*depth-complexity*), o que, como veremos, prejudica alguns algoritmos. Outra dificuldade é o grande custo de

geometria, nem sempre amenizado por descarte em relação ao volume de visão (*frustum culling*), já que freqüentemente toda a cena é visível tanto para a fonte de luz quanto para a câmera.

Por outro lado, em muitas aplicações a cena é estática, de modo que um mapa de sombras simples não precisa ser recriado continuamente a cada quadro. Além disso, não precisamos nos preocupar com fontes de luz onidirecionais, apenas direcionais ou, no máximo, luzes *spot*. Finalmente, apesar de ser sempre desejável a renderização em tempo real, muitas vezes um requerimento de tempo interativo é suficiente.

1.3. Contribuições e organização do trabalho

Neste trabalho, fazemos uma extensa análise do chamuscamento e do serrilhamento, os dois principais problemas associados a algoritmos de mapeamento de sombras no contexto de visualização de modelos CAD. Estudamos também várias técnicas que podem ser utilizadas para melhorar a qualidade das sombras, verificando quais se adequam melhor a esse tipo de aplicação. Ao longo do estudo, propomos três adaptações que podem ser feitas aos algoritmos para melhorar a eficiência e a qualidade das sombras: o cálculo de amostras virtuais baseada inteiramente em programas de fragmento (Seção 3.1.2), o uso de um parâmetro n'_{LiSPSM} generalizado para diferentes configurações de luz e câmera (Seção 4.2.2), e um esquema de particionamento em z adaptativo (Seção 4.3.2). Todas as técnicas foram implementadas e testadas na visualização de modelos CAD.

O trabalho está organizado da seguinte forma: no Capítulo 2, revisamos os progressos recentes e as principais técnicas já propostas para tratar cada problema do mapeamento de sombras ou estender o algoritmo. No Capítulo 3, investigamos técnicas para a obtenção de sombras de boa qualidade sem alterar a amostragem da cena pelo mapeamento. Essas técnicas tentam eliminar o efeito de chamuscamento e ocultar o serrilhamento aplicando algum tipo de filtro às bordas das sombras. No Capítulo 4, por outro lado, analisamos em detalhes o efeito de serrilhamento e investigamos técnicas que alteram a amostragem da cena,

reparametrizando e particionando o mapa de sombras de modo a melhorar o aproveitamento de sua resolução. O Capítulo 5 explica detalhes relativos à implementação realizada e, em seguida, apresenta e discute os resultados obtidos, avaliando os algoritmos em critérios como a facilidade de implementação, a qualidade visual obtida e a eficiência conseguida (medida em FPS, quadros-por-segundo). Por fim, no Capítulo 6, conclui-se o trabalho e sugerem-se direções de trabalhos futuros. Os Apêndices A e B apresentam ainda algumas derivações matemáticas omitidas no texto.