Uma Versão Rápida do Algoritmo Scan-line para Rendering de Sólidos

Carlos Neves Lenz César [1]
Maria Cristina Ferreira de Oliveira [2]

USP - Universidade de São Paulo
ICMSC - Instituto de Ciências Matemáticas de São Carlos
Caixa Postal 668, 13560-970 - São Carlos - SP
[1] cnlcesar@icmsc.sc.usp.br
[2] ferreira@icmsc.sc.usp.br

Abstract. This paper describes a modified version of a scan-line algorithm suited for fast rendering of poliedral manifold solid models. The implementation explores inherent properties of such solids in order to reduce execution times. The algorithm is being integrated into the visualization module of a solid modeling system that keeps a B-Rep structure as the main internal representation.

Introdução

O processo de *Rendering* consiste na formação de uma imagem bidimensional iluminada (*shaded*) a partir da projeção de uma cena (coleção de objetos e seu ambiente) tridimensional armazenada em uma base de dados. O objetivo do *rendering* é obter uma imagem de boa qualidade e de fácil interpretação. Para isto é necessário remover as superfícies ocultas dos objetos e calcular a luz emitida pelos objetos ao observador. O resultado esperado é uma imagem com aparência próxima da real.

Um dos maiores problemas no *rendering* é o esforço computacional exigido, principalmente para a remoção de superfícies ocultas. Com isto, técnicas de remoção de superfícies ocultas têm exigido intenso esforço em pesquisas com o objetivo de encontrar algoritmos mais eficientes, além de arquiteturas dedicadas.

Existem vários algoritmos para remoção de superfícies ocultas, e provavelmente os mais utilizados são: *Ray-tracing*, *Scan-line* e *Z-buffer* [Rogers (1985)] [Foley et al. (1990)]. O *Scan-line* é um dos mais rápidos no caso geral, pois se utiliza de várias coerências que economizam cálculos além de possibilitar a incorporação de efeitos visuais simples, permitindo a formação de imagens de boa qualidade.

Este trabalho descreve a construção de um módulo de visualização baseado no algoritmo *Scan-line* o qual deve ser integrado ao $(SM)^2$ - Sistema de Modelagem de Sólidos Multirepresentacional. O $(SM)^2$ é um modelador de sólidos voltado para aplicações em ensino e pesquisa, que está em desenvolvimento no Instituto de Ciências Matemáticas de São Carlos (ICMSC) da Universidade de São Paulo (USP) [Magalhães et al. (1994)]. O sistema suporta diferentes técnicas de descrição de sólidos, e os modelos gerados representados na forma B-Rep (*Boundary Representation*) [Mäntylä (1988)].

A versão do algoritmo utilizada no módulo de visualização incorpora algumas otimizações que permitem acelerá-lo, as quais exploram propriedades dos modelos de sólidos poliedrais. O módulo permite a visualização de sólidos utilizando um modelo de iluminação descrito em [Rogers (1985)], o qual emprega uma função de iluminação com três termos, referentes às contribuições de luz ambiental, difusa e especular. Diferentes técnicas de *shading* [Rogers (1985)], [Foley (1990)] poderão ser utilizadas na visualização, estando o *Flat shading* operacional, e *Gouraud shading* e *Phong shading* em desenvolvimento.

Este trabalho está organizado da seguinte maneira: primeiramente, são explicadas as características dos sólidos que são exploradas para acelerar o algoritmo de remoção de superfícies ocultas. A seguir, descreve-se o funcionamento do algoritmo *Scan-line* genérico, bem como uma implementação específica. Na próxima seção, o algoritmo utilizado para a visualização dos sólidos no $(SM)^2$ é descrito, e finalmente são feitas algumas observações sobre os resultados obtidos.

Propriedades dos Sólidos

Uma B-Rep consiste de uma representação computacional que armazena informações topológicas (faces, arestas, vértices, etc., e suas relações de adjacência) e geométricas relativas à superfície limitante de um sólido. Algoritmos de visualização e remoção de superfícies ocultas podem explorar diretamente estas informações.

Uma propriedade dos sólidos é que sua superfície limitante envolve todo o seu volume, definindo assim o lado interior e o exterior [Requicha (1980)]. A representação para descrever os sólidos utilizada deve ser capaz de identificar o lado interior e exterior de cada face.

Os sólidos mencionados neste trabalho são

poliedrais, ou seja, são compostos apenas por faces planares e arestas retilíneas. Isto significa que um sólido contendo uma superfície limitante "curva" é representado de forma aproximada por várias faces planas.

Além disso, a fronteira dos sólidos representados na B-Rep empregada pelo $(SM)^2$ são variedades de dimensão 2, ou *2-manifolds* [Mäntylä (1988)]. Assim, cada aresta de uma face é compartilhada por exatamente duas faces, e além disso, as faces do sólido não se auto-interceptam, a não ser em arestas comuns.

Funcionamento Básico do Algoritmo Scan-line

O algoritmo *Scan-line* processa a cena na ordem da linha de varredura do dispositivo, gerando a imagem de cima para baixo (ou de baixo para cima). A remoção de superfícies ocultas é realizada em cada linha de varredura da imagem. O algoritmo convencional trabalha com faces de forma independente, ou seja, faces que não necessariamente pertencem a sólidos.

O primeiro passo do algoritmo reduz o problema de três para duas dimensões. Este passo encontra os segmentos de linha das faces que são projetadas em cada linha de varredura. Para isto, um plano de varredura é definido pela posição do observador e uma linha de varredura. A interseção entre o plano de varredura e a cena tridimensional define uma janela bidimensional, como mostrado na figura 1, e nesta janela será resolvido o problema que é então reduzido a decidir que segmento de linha é visível para cada ponto na linha de varredura, o que é feito no segundo passo.

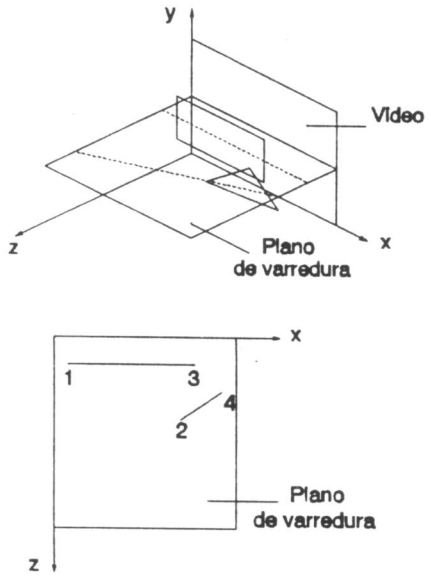

Figura 1 - Plano de varredura.

No segundo passo, pode-se utilizar o método denominado *Spanning Scan-line* [Rogers (1985)]. Neste algoritmo introduz-se o conceito de *spans* (blocos de pixels). Os *spans* são obtidos dividindo a linha de varredura nas extremidades e cruzamentos dos segmentos de retas. As partes obtidas são chamadas *spans* (figura 2). A solução do problema de remoção de superfícies ocultas é reduzida à seleção do segmento visível em cada *span*. Apenas três tipos de *spans* são possíveis:

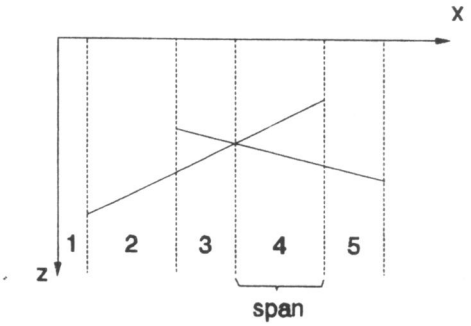

Figura 2 - Exemplo de *spans*

- O *span* é vazio (*span* 1). Neste caso, a cor de fundo é exibida.
- O *span* contém apenas um segmento (*spans* 2 e 5). Neste, os atributos da face correspondente ao segmento são exibidos neste *span*.
- O *span* contém mais de um segmento (*spans* 3 e 4). O nível (distância ao observador) de cada segmento é calculado. O segmento com o menor nível é o segmento visível em todo o *span* e os atributos deste segmento são exibidos no *span*.

O Algoritmo Original do Scan-line

Uma implementação do algoritmo *Scan-line* utilizando a técnica *Spanning Scan-line* [Foley et al. (1990)] é descrita nesta seção. O algoritmo básico é uma extensão do algoritmo de conversão matricial de polígonos. A diferença principal é que não é executado sobre um único polígono, mas sobre um conjunto de polígonos.

Nesta implementação são utilizadas várias formas de coerência para otimizar o algoritmo, como por exemplo coerência de linhas de varredura e coerência de arestas. Também é evitado o uso de variáveis em ponto flutuante, sendo utilizados nas principais operações números inteiros.

O algoritmo utiliza as seguintes estruturas de dados:

- **Tabela de Arestas** - contém todas as arestas não horizontais de todos os polígonos projetados no plano de visão. As arestas são ordenadas em *buckets* com base na menor coordenada *y* das arestas. Para cada aresta são armazenados: a coordenada *x* correspondente a menor coordenada *y* da aresta, a coordenada *y* máxima, o

incremento de x e um identificador do polígono na qual a aresta está contida.

• **Tabela de Arestas Ativas** - contém as arestas da Tabela de Arestas interceptadas pela linha de varredura y atual. Esta é atualizada a cada mudança de linha de varredura e sempre se mantém ordenada pela coordenada x da interseção da aresta com a linha de varredura.

• **Tabela de Polígonos** - contém informações de cada polígono (face). Para cada polígono são armazenados os coeficientes da equação do plano, informação de cor e intensidade e um flag booleano "in-out" inicializado como "falso" e utilizado durante o processamento da linha de varredura.

Apresentamos a seguir um resumo do algoritmo:

1. Cria a Tabela de Polígonos e a Tabela de Arestas.
2. Inicializa a Tabela de Arestas Ativas como vazia.
3. Para cada linha de varredura y:
   - 3.1 Transfere as arestas do bucket y na Tabela de Arestas para a Tabela de Arestas Ativas, reordenando-a em x.
   - 3.2 Traça os pixels da linha de varredura conforme o critério definido na técnica Spanning Scan-line. A Tabela de Arestas Ativas determina os spans e a Tabela de Polígonos contém os atributos que serão exibidos e outras informações que determinam qual o segmento visível em cada span.
   - 3.3 Retira da Tabela de Arestas Ativas as arestas que não pertencem a próxima linha de varredura.
   - 3.4 Atualiza os valores x das arestas da Tabela de Arestas Ativas para a nova linha de varredura.

Algoritmo Scan-line para Sólidos

A versão do algoritmo descrita anteriormente permite visualizar qualquer conjunto de faces (polígonos) planas, não sendo necessário que estas faces definam sólidos. O algoritmo proposto neste trabalho tira proveito das propriedades dos sólidos. As propriedades dos sólidos utilizadas e as vantagens em aproveitá-las são descritas a seguir:

• **As faces dos sólidos possuem lado interno e externo** - é obvio que é visível diretamente apenas o lado externo da face. O lado interno só será visível se o observador estivesse no interior do sólido, se o sólido for transparente ou se a imagem deste sólido for refletida por outros sólidos. Em uma visualização simples, estes casos não ocorrem. Uma **face de trás** possui o seu lado interno voltado ao observador e pode ser eliminada no processo de visualização do objeto. Esta técnica é chamada de **Remoção de Faces de Trás** (*Back-Face Culling*) [Foley et al. (1990)], [Hearn et al. (1986)].

Para identificar as faces de trás, obtém-se um vetor normal para cada face, o qual está orientado para o lado externo do objeto. Uma face de trás pode ser identificada pelo resultado do produto escalar entre o vetor normal à face e o vetor que vai do observador a um ponto na face. Se o produto escalar é não negativo, tem-se uma face de trás. Se o vetor normal à face estiver em coordenadas do dispositivo o produto escalar pode ser substituído pela verificação da coordenada z do vetor normal.

Este processo não remove todas as faces ocultas dos sólidos. Em geral, a técnica de Remoção de Faces de Trás pode ser utilizada para remover em torno da metade das faces de uma cena, e então é utilizado outro algoritmo de remoção de superfícies ocultas.

Esta técnica pode ser empregada no algoritmo *Scan-line* durante a criação da Tabela de Polígonos, bastando não inserir as faces de trás nesta tabela. O custo para identificar as faces de trás é mínimo, pois ele é simples e de complexidade linear. Quando utilizamos este processo, o tamanho das tabelas do *Scan-line* é reduzido aproximadamente pela metade, e conseqüentemente, é reduzido pela metade o tempo para processá-las. Outro ganho na eficiência é devido a redução do número de segmentos nos *spans*, reduzindo assim o custo para identificar o segmento visível nos *spans*.

Como esta técnica não deve ser utilizada em alguns casos, como quando se utiliza transparência, sua utilização deve ser opcional.

• **Faces vizinhas possuem uma aresta em comum** - a Tabela de Arestas utilizada no *Scan-line* descrito anteriormente possui todas as arestas de todas as faces, tendo assim arestas duplicadas no caso de sólidos, pois cada aresta é inserida duas vezes, uma para cada face. Os dados da Tabela de Arestas podem ser alterados acrescentando outro identificador de polígono. Assim cada aresta poderá pertencer a duas faces.

Esta alteração permite a redução do tamanho da Tabela de Arestas aproximadamente pela metade, reduzindo também o tempo para processá-la. Outro ganho em eficiência é quando utiliza-se do fato de que, se uma aresta pertence a duas faces, elas são vizinhas e portanto estas faces são provavelmente sucessoras nos *spans*. Com isso, não é necessário, em muitos casos, comparar níveis das faces no ponto da sucessão.

• **Faces de sólidos nunca se cruzam** - utilizando-se desta propriedade pode-se simplificar a utilização e criação dos *spans*. Ela garante que não há cruzamentos de segmentos dos *spans* e os *spans* são identificados apenas pelas extremidades dos segmentos. Assim, podemos garantir que um segmento é visível até o seu

término ou até o início de outro segmento.

Resultados Obtidos

A implementação do algoritmo permitiu um *rendering* rápido de sólidos sem perda na qualidade da imagem. A desvantagem é a impossibilidade de visualizar objetos que não sejam sólidos poliedrais.

Atualmente o visualizador permite a geração de imagens simples, com uma única fonte de luz direcional utilizando a técnica de *Flat Shading*. Estão em fase de implementação técnicas de *shading* mais sofisticadas como *Phong-shading* e *Gouraud-shading*, e técnicas para a obtenção de sombras e transparência.

Foram feitos alguns testes comporativos para verificar o desempenho do algoritmo que incorpora as otimizações sugeridas. A tabela abaixo ilustra o tempo gasto (em segundos) na execução do algoritmo, onde,

N.Faces - é o número de faces da figura;

Alg.1 - é o algoritmo na versão original;

Alg.2 - é o algoritmo otimizado sem remoção prévia de faces de trás;

Alg.3 - é o algoritmo que incorpora todas as otimizações sugeridas.

Figura 3, 4 e 5 - são as figuras mostradas no final deste trabalho (400 x 600 pixels).

	N.Faces	Alg.1	Alg.2	Alg.3
Figura 3	128	0,800	0,605	0,290
Figura 4	624	1,295	0,945	0,500
Figura 5	1992	2,729	1,713	0,759

Observa-se na tabela que as otimizações sugeridas permitiram uma visualização que exige em torno de 33% do tempo necessário para a execução da versão original do algoritmo. O algoritmo apresentado está sendo implementado na linguagem C numa estação de trabalho Sun SparcStation II.

Referências

J. D. Foley et al, *Computer Graphics: Principles and Practice*, 2nd edition, Addison-Wesley, 1990.

D. Hearn, M. P. Baker, *Computer Graphics*, Prentice-Hall, 1986.

A. L. C. C. Magalhães et al., Desenvolvimento de um modelador de sólidos com núcleo BRep e técnicas de de descrição por varredura e semi-espaços, *paper submetido ao VII SIBGRAPI*, 1994.

M. Mäntylä, *An Introduction to Solid Modeling*, Computer Science Press, 1988.

A. A. G. Requicha, Representations of solid objects-theory, methods, and systems. *ACM Computing Surveys*, 12(4):437-464, Dec. 1980

D. F. Rogers, *Procedural Elements for Computer Graphics*, McGraw-Hill, 1985.

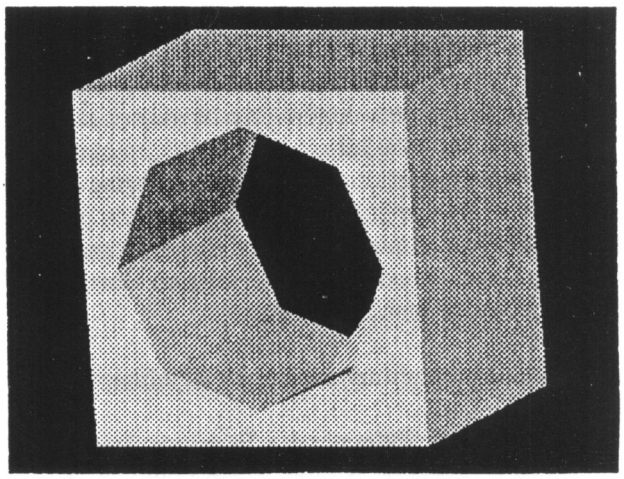

Figura 3 - Cubo vazado

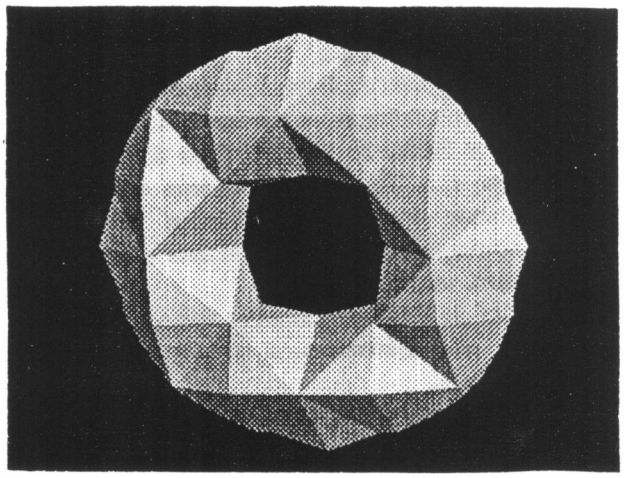

Figura 4 - Toro

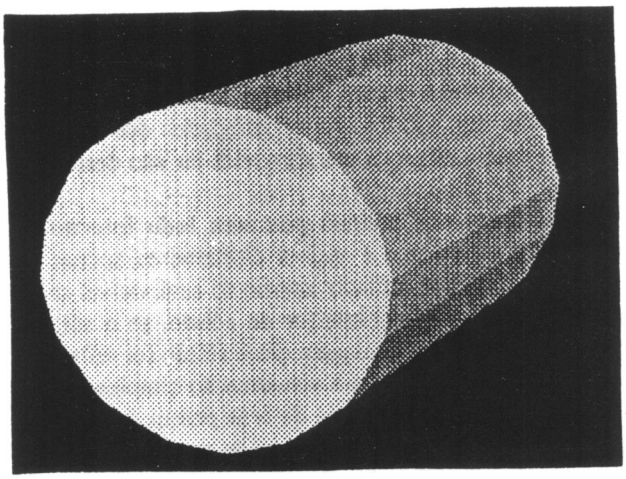

Figura 5 - Cilindro