Renderização de Objetos Translúcidos Discretos com Aproximação da Superfície por Interpolação Bidimensional

Maliska Junior, C. R.

SINMEC - Laboratório de Simulação Numérica em Mecânica dos
Fluidos e Transferência de Calor
Depto. de Engenharia Mecânica - U.F.S.C.

1. SUMÁRIO

A renderização é um processo intensamente utilizado nos sofwares de visualização tridimensional. A mesma consiste numa simplificação do processo de ray-tracing e possibilita uma visualização mais realística do objeto, em relação àquela conseguida com o processo de hidden-lines. No presente trabalho é mostrada uma técnica de renderização de objetos translúcidos discretos, ou seja, que possibilitam a passagem de luz pelos mesmos, com a aproximação da superfície por interpolação bidimensional. Juntamente com o mesmo é mostrada uma técnica de visualização de objetos em três dimensões, bem como de hidden-lines e tratamento de cores.

2. INTRODUÇÃO

Quando se procura visualizar um objeto tridimensional, sem a preocupação de fotorealismo mas objetivando uma boa caracterização 3D do mesmo, o processo de renderização apresenta-se como uma alternativa econômica quando comparada a técnica do ray-tracing. Nesta última técnica todos os pontos da tela são utilizados para cálculo de textura, brilho e cor, constituindo-se em um processo demorado e custoso.

Em muitas situações de engenharia a boa visualização do objeto, sem a preocupação da representação da textura e outras propriedades pontuais, é suficiente a renderização, que trabalha apenas com os pontos superficias do objeto, em um processo extremamente

simples e rápido. Com este objetivo, o presente trabalho mostra um processo para a renderização de objetos translúcidos discretos, onde a aproximação da superfície é feita por interpolação bidimensional. A metodologia básica consiste no cálculo da incidência de luz sobre o objeto, um processo geométrico tridimensional de fácil implementação.

Diversas rotinas básica são necessárias previamente ao processo de renderização, tais como, rotinas de visualização do wire-frame, hidden-lines e outras. Estas rotinas também são descritas, mesmo brevemente, para o melhor entendimento do processo de renderização.

A rotina de renderização aqui apresentada está associada a descrição discreta de objetos, geralmente obtidas com geradores de malhas empregados em modelações numéricas de problemas físicos.

3. ROTINAS BÁSICAS

3.1. Definição do Objeto a ser Renderizado

Os objetos que serão aqui renderizados são todos definidos por quadriláteros não entrelaçados, sendo necessário o fornecimento das coordenadas (x,y,z) dos vértices. Estes quadriláteros estão dispostos de uma forma organizada, na qual cada um possue somente 4 vizinhos (exceto os elementos de fronteira), formando uma malha denominada estruturada. Em função disso, trabalha-se somente com pontos, retas e superfícies elementares definidas por 4 pontos. Diferentes malhas estruturadas, definindo diferentes superfícies, podem ser combinadas de forma a formarem objetos complexos.

Os pontos que definirão o objeto devem ser escritos em uma mesma base, ortogonal e unitária. Esta base é definida como a base do objeto e é dada por

$$B_o = \begin{pmatrix} \vdots & \vdots & \vdots \\ v_{o1} & v_{o2} & v_{o3} \\ \vdots & \vdots & \vdots \end{pmatrix} \tag{3.1}$$

onde $(B_o)^2 = I$.

3.2. Rotinas de Câmera para Visualização do Wire-Frame

Para a visualização de um objeto tridimensional, é necessário, como condição mínima, a indicação da posição em que se encontra o observador, da direção de observação e do horizonte. A indicação de luzes, textura superficial e outras informações são acréscimos que irão dar um maior tom de realismo à figura.

Conforme comentado, os objetos elementares que estamos trabalhando são formados por pontos. Para a definição da posição dos pontos no espaço, trabalha-se com uma segunda base, também ortogonal e unitária, aqui definida de base de visualização, dada por

$$B_r = \begin{pmatrix} \vdots & \vdots & \vdots \\ v_{r1} & v_{r2} & v_{r3} \\ \vdots & \vdots & \vdots \end{pmatrix} \tag{3.2}$$

Esta base relaciona o observador e a tela, ou seja, transformando o objeto da base em que ele foi escrito para a base de visualização, tem-se as coordenadas dos pontos em relação à base da tela. Assim, basta desenhar-se o objeto com as coordenadas obtidas. Para transformar o objeto para a base de visualização, basta multiplicar os vetores que definem os pontos, escritos na base do objeto, pelos três vetores coluna que formam a matriz da base de visualização, como

$$\begin{pmatrix} \vdots \\ P \\ \vdots \end{pmatrix}_{B_t} = \begin{pmatrix} \vdots & \vdots & \vdots \\ v_{r1} & v_{r2} & v_{r3} \\ \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \vdots \\ P \\ \vdots \end{pmatrix}_{B_o} \tag{3.3}$$

3.3. Rotinas de Hidden-lines

Feita a definição da posição do objeto na tela, com todos os valores (x,y,z) dos pontos formadores, precisa-se agora dar volume ao objeto. Para isto, é necessário um processo de hidden-lines.

Após terem sido implementados alguns métodos de hidden-lines, chegou-se a conclusão que, para este caso, a utilização do processo z-buffer é o mais adequado, em função da facilidade de implementação e da velocidade de processamento.

Como deseja-se aplicar o processo de hidden-lines sobre um objeto formado apenas por quadriláteros, pode-se definir dois processos derivados da técnica z-buffer, o hidden-lines grosseiro e o refinado. Os dois diferem em relação ao processamento de intersecções e da distorção do objeto.

O processo de hidden-lines grosseiro consiste no mapeamento de todos os quadriláteros elementares em um vetor com dimensão igual ao número de quadriláteros. Cada um deles possue uma posição neste vetor e um valor relacionado a ele, que é a coordenada z do seu ponto médio dada pela média aritmética da coordenada z dos quatro vértices. Um método de organização em ordem decrescente da coordenada z é aplicado à este vetor (o método quick-sort invertido é o mais adequado em função da sua velocidade) e a plotagem dos elementos na ordem deste é efetuada. O resultado é a impressão dos elementos de trás para frente. Quando completado o processo de plotagem, o objeto está montado.

O método descrito não processa a intersecção de quadriláteros e, quando a mesma ocorrer, como é o caso da intersecção de duas superfícies definidas por duas malhas estruturadas diferentes, conforme Fig. 3.1, haverá um erro na figura. Além deste, o ponto médio nem sempre indica se um quadrilátero está na frente ou atrás de outro, e por isso, quando a figura contiver elementos com discrepâncias muito grandes de tamanho, poderá

ocorrer um erro de criação da imagem. Mesmo assim, os resultados são muito bons para um processo tão rudimentar e de tão simples implementação.

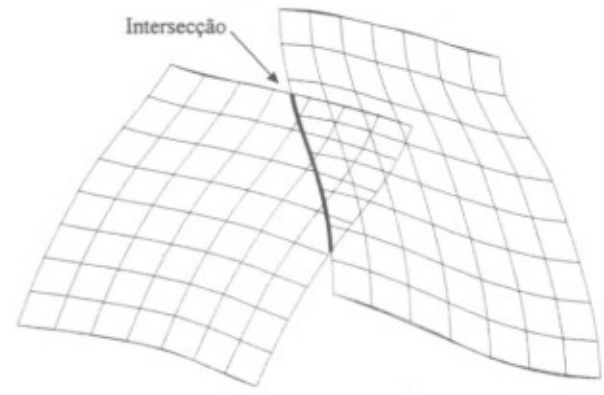

Figura 3.1: Intersecção de duas superfícies

Já o processo de hidden-lines refinado consiste na criação de uma matriz z-buffer do tamanho da tela em que se criará a imagem. Esta matriz deve, inicialmente, conter somente valores nulos. Com isso, começa-se o processo de plotagem dos quadriláteros, considerando-se que para cada ponto do elemento desenhado, deve-se escrever o valor da coordenada z do ponto na matriz z-buffer, na posição correspondente àquela em que o ponto foi desenhado. Com isso, esta matriz indicará a distância de cada ponto da tela já desenhado ao observador. Deve-se, no entanto, notar que quando for plotado um ponto do quadrilátero deve-se, primeiro, verificar se a posição em que ele irá ser plotado já possue um ponto ou não. Se já possuir, verifica-se se o valor da coordenada z do ponto já plotado é maior ou menor que o valor na coordenada z do ponto a ser plotado. Caso positivo o novo ponto está mais próximo do observador e deve ser plotado. Caso contrário, nada faz-se, pois o ponto que iria ser plotado está oculto por uma outra face.

Com isso, o processamento de intersecção entre as superfícies é realizado e não ocorrem erros de hidden-lines em função da distorção do elemento, como ocorriam no caso do hidden-lines grosseiro. No entanto, o processo de hidden-lines grosseiro é bem mais rápido que o refinado, ainda mais se a computador possuir um acelerador de polígnos razoavelmente veloz, opção que o hidden-lines refinado não pode utilizar em função de trabalhar somente com a plotagem de pontos.

3.4. Propriedades do Objeto Renderizado

O processo que será aqui apresentado consiste no cálculo da incidência de luz sobre um objeto qualquer definido anteriormente. Este objeto possui as propriedades de um objeto translúcido do ponto de vista de que sua renderização não originará sombras. Para efeitos

de visualização é lógico que o mesmo deverá ter capacidade de refletir a luz incidente, que é exatamante o processo em descrição.

3.5. Tratamento das cores para o objeto renderizado

Quando um processo de renderização é aplicado sobre algum objeto, um mapa de cores deve ser definido de maneira a poder desenhar as cores adequadas conforme a incidência de luz calculada. Em função de, normalmente, os vídeos serem de 8-bits, que possibilita a apresentação de somente 256 cores simultâneamente, armazenadas em um vetor que é o mapa de cores do vídeo, o objeto aqui renderizado foi definido como sendo inteiramente de uma só cor.

Esta cor é definida na escala RGB por (a,b,c), e o mapa de cores gerado deve iniciar com a cor preta (0,0,0) (posição inicial do vetor), variar até a cor definida (a,b,c) (posição 128 do vetor, o centro), e igualmente até o branco (256,256,256) (fim do vetor). Com isto, teremos a impressão da ausência de luz (escuro), até a cor do objeto e encaminhando-se para o incidência total de luz.

Esta é a escala de cores utilizada na renderização do objeto. Alterações na mesma podem ser feitas a critério do usuário, para aumentar ou diminuir a incidência de luz, bem como a cor da luz e da ausência da mesma. A escala de cores, então, varia de 0 a 256, e a intensidade de luz de 0 a 1. Assim, quando um ponto for desenhado, se a sua intensidade de luz for 0, a sua cor será 0, e se a intensidade de luz for 1, sua cor será 256. Os demais valores são encontrados por interpolação linear.

4. O PROCESSO DE RENDERIZAÇÃO PROPRIAMENTE DITO

Serão apresentados duas variantes do método, uma para um resultado discreto, e outro para um resultado contínuo. O resultado discreto provém de um tratamento isolado de cada quadrilátero. Já o resultado contínuo provém de um processo que relaciona o quadrilátero com seus vizinhos, aproximando a incidência da luz por uma função de interpolação bidimensional.

4.1. O Processo Discreto

O processo discreto de renderização consiste em calcular a intensidade de luz para cada um dos quadriláteros existentes na figura. Assume-se, então, que todo o quadrilátero possue a mesma incidência de luz. Desta maneira, quando aplicado o processo de hidden-lines, pinta-se o elemento com a cor correspondente à sua incidência de luz. Este processo proporciona um resultado descontínuo, conforme pode ser visto na figura 5.1, 5.3, 5.5 e 5.7, ou seja, existem saltos entre as cores atribuidas à superfície do objeto, o que fica desagradável ao observador se o objeto possue poucos elementos de formação.

4.2. O Processo Contínuo

O processo contínuo de renderização consiste em fazer o processo discreto não pintando o elemento todo com a mesma cor, mas sim empregando uma função interpolação bidimensional para as cores. Esta função interpolação[1] é agora descrita.

A Fig. 4.1 apresenta os pontos onde se dispõe dos valores da intensidade da luz calculados. Como são conhecidos os vizinhos do ponto P, N, S, E, W, NW, NE, NS, SE, podemos calcular os valores nos pontos A, B, C e D. Com isso, cria-se um novo quadrilátero onde será aplicada a função interpolação.

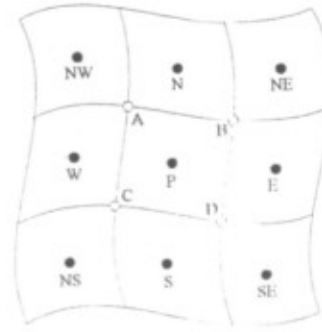

Figura 4.1: Quadriláteros elementares e os pontos de armazenamento da incidência de Luz

A função interpolação consiste em, conhecidos os valores nos pontos A, B, C, e D, como mostrado na Fig. 4.2, interpolar-se linearmente os valores para o ponto E e F, e novamente para o ponto G. Assim, pode-se obter os valores da intensidade da luz em todos os pontos, sem aplicar a função que calculo esta intensidade. Com este processo, está-se, também, aplicando uma função interpolação sobre a coordenada z do objeto, o que cria um superfície contínua e derivável, já que a mesma é uma composição de funções deriváveis.

4.3. Processo para o Cálculo da Intensidade Luminosa

O processo para o cálculo da intensidade luminosa é uma função que retorna um valor entre 0 e 1, que representa ausência total e incidencia total de luz em um ponto de uma determinada geometria.

Como o processo aqui apresentado é aplicado sobre um objeto translúcido, a intensidade de luz depende somente do angulo entre a direção de visão do observador e a direção de reflexão do raio luminoso. Assim, a função depende somente do quadrilátero em questão,

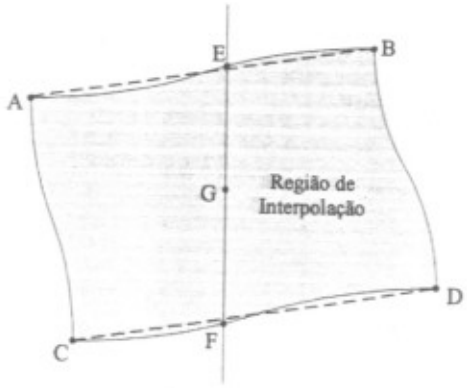

Figura 4.2: Esquema da função interpolação bidimensional

dos pontos de luz e da posição sobre o quadrilátero em que se deseja calcular a intensidade de luz.

Para isto basta calcular o angulo entre a direção de visão do observador (perpendicular à tela) e a reta refletida pelo quadrilátero que liga o ponto de luz com o ponto onde se deseja calcular a intensidade de luz. A equação para este cálculo é

$$
\begin{aligned}
x_p &= \frac{A^2 x_l + x_c\,(B^2 + C^2) + AB\,(y_l - y_c) + AC\,(z_l - z_c)}{A^2\,B^2\,C^2} \\
y_p &= \frac{AB\,(x_l - x_c) + B^2\,(y_l - y_c) + BC\,(z_l - z_c)}{(A^2\,B^2\,C^2) + y_c} \\
z_p &= \frac{AC\,(x_l - x_c) + CB\,(y_l - y_c) + C^2\,(z_l - z_c)}{(A^2\,B^2\,C^2) + z_c}
\end{aligned}
\tag{4.1}
$$

onde a A, B, C, e D são os coeficientes da equação do plano $Ax + By + Cz + D = 0$, que é uma aproximação da superfície que passa pelos quatro pontos que definem o quadrilátero, (x_l, y_l, z_l) são as coordenadas do ponto de luz e (x_c, y_c, z_c) são as coordenadas onde deseja-se calcular a incidência de luz. Com isso, usam-se os valores de x_p, y_p e z_p em

$$
\begin{aligned}
x_r &= 2x_p - x_l \\
y_r &= 2y_p - y_l \\
z_r &= 2z_p - z_l
\end{aligned}
\tag{4.2}
$$

que, substituidos em,

$$
ang = arccos\left(\frac{z_r - z_c}{sqrt(x_r - x_c)^2 + (y_r - y_c)^2 + (z_r - z_c)^2}\right) \tag{4.3}
$$

determina o ângulo entre o raio refletido e a direção de visão do observador. Para obter-se o coeficiente de incidência de luz, basta dividir este ângulo por 90.

5. Resultados

Nas duas próximas páginas são apresentados alguns resultados obtidos com o método anteriormente apresentado. As Figs. de 5.1 a 5.8 são objetos renderizados pelos métodos aqui descritos. As Figs. 5.1, 5.3, 5.5 e 5.7 foram obtidas usando o método de renderização discreto, e as Figs. 5.2, 5.4, 5.6 e 5.8 foram obtidas aplicando-se o método de renderização contínuo. As imagens de 1200x900 pixeis, obtidas em uma SPARC Station 10 com processador P42, gastaram 4.9s, 18.5s, 8.5s, 25.5s, 3.5s, 13.8s, 11.0s, e 31.0s de CPU, respectivamente.

6. Conclusões

Um método de renderização de objetos discretos translúcidos foi apresentado, além de outros métodos de visualização tridimensional de objetos e hidden-lines. O método de aproximação da superfície por função interpolação bidimensional mostrou-se de fácil implementação, bem como o método discreto de renderização, e apresentou resultados muito bons, obtendo um aspecto visual bastante agradável e fiel à geometria.

Os resultados com relação ao custo/benefício também são compensadores, comparando com os métodos convencionais de renderização. A facilidade de implementação e simplicidade de todo o método é um fator extremamente positivo.

7. Referências Bibliográficas

1. Maliska Jr., C. R. e Dihlman, A., "Visualizador Tridimensional Para Campos Escalares e Vetoriais", Anais do XII CILAMCE - Congresso Ibero Latino-Americano Sobre Métodos Computacionais Para a Engenharia, Porto- Alegre - RS, novembro de 1992.

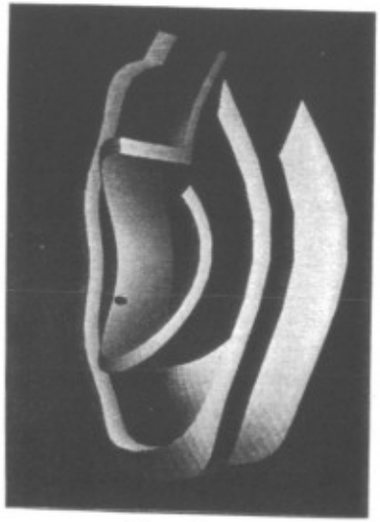

Figura 5.1

Figura 5.2

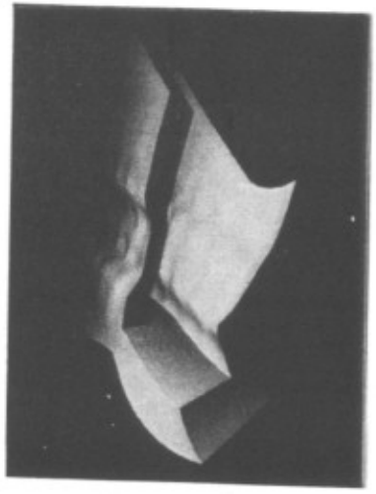

Figura 5.3

Figura 5.4

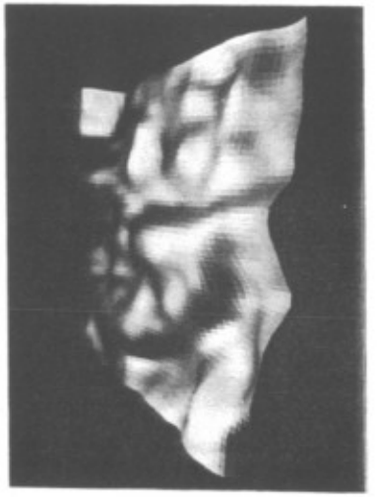

Figura 5.5

Figura 5.6

Figura 5.7

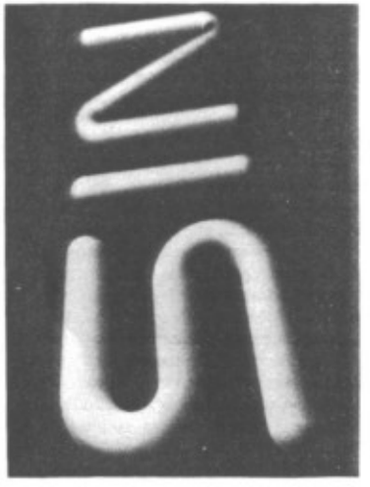

Figura 5.8