Determinar os limites de *viewport* circular a partir de uma função hashing

Alex F. de Araújo, Aledir Silveira Pereira, Norian Marranghello
Depto de Ciências de Computação e Estatística, IBILCE, UNESP
15054-000, São José do Rio Preto, SP, Brasil
E-mail: fa.alex@gmail.com, aledir@ibilce.unesp.br, norian@ibilce.unesp.br

Acrísio José do Nascimento Júnior
Depto de Ciências da Computação, UFG / CaC
75705-200, Catalão, GO, Brasil
E-mail: acrisio@catalao.ufg.br

João Manuel R. S. Tavares
Depto de Eng. Mecânica e Gestão Industrial (DEMEGI), Fac. de Eng. da Faculdade do Porto (FEUP)
Instituto de Engenharia Mecânica e Gestão Industrial (INEGI)
4200-465, Porto, Portugal
E-mail: tavares@fe.up.pt

RESUMO

Encontrar os limites de uma janela de visualização (viewport) é o processo mais exigente computacionalmente do estágio de clipping (recorte) do pipeline de visualização gráfica, [2], [3]. O processamento torna-se mais exigente quando aplicado a imagens matriciais (como do tipo bitmap). Neste caso, cada pixel deve ser analisado individualmente, comparando sua distância ao centro da área visível. Este trabalho faz uso dos conceitos de hashing para encontrar os limites do plano de recorte circular, definindo o contorno deste plano diretamente e evitando comparar os pixels inválidos, aumentado assim a sua eficiência.

A estrutura hashing possui uma abordagem de busca diferente das listas, filas e árvores. Ao contrário do que ocorre nas demais estruturas, onde a busca por um determinado elemento é feita através de um percurso, comparando cada elemento visitado, no hashing o acesso ao conteúdo ocorre por meio de uma chave única a qual é obtida através de uma função matemática chamada função hashing. A busca ocorre através do uso de tabelas de indexação, onde cada posição é associada a uma chave. Desta forma, ao conhecer as chaves torna-se possível acessar as posições desejadas na tabela sem precisar percorrê-la. Se chaves diferentes forem transformadas em índices distintos a função é perfeita, [1].

Considerando um vetor com r posições, onde r é o raio da circunferência que forma a janela de visualização, é possível formular uma função hashing perfeita h, que preenche este vetor com os parâmetros utilizados para buscar os pixels da matriz que se encontram no limite da região visível. O conjunto de chaves usado é definido pelos números inteiros que pertencem ao intervalo fechado de 0 (zero) até r. A região visível é limitada por uma circunferência, logo o seu centro foi considerado como a origem do sistema de coordenadas (posição (0, 0)) da cena. Assim, a equação que descreve a janela de corte é dada por:

$$y = (r^2 - x^2)^{1/2} \quad . \qquad (1)$$

Considerando apenas os valores positivos de x e y, tem-se uma função que converte um valor de entrada em apenas uma saída. Portanto, nossa função perfeita h, fica da seguinte forma:

$$h(x) = (r^2 - x^2)^{1/2}, \qquad (2)$$

com $x \geq 0$.

A circunferência é um polígono simétrico, com isso, para cada coordenada x tem-se duas ordenadas (y e –y) e para cada x positivo tem-se seu espelho do lado negativo, ou seja, para todo x maior que 0 (zero) e menor que r, existe um –x menor que 0 (zero) e maior que -r. Portanto, cada parâmetro x do vetor gerado pela função hashing adotada pode ser usado para localizar um pixel de contorno em cada quadrante do plano de corte.

Ao usar o vetor de parâmetros definido pela função hashing adotada, torna-se possível efetuar uma busca direta pelos pixels que limitam o plano de corte circular em uma imagem matricial, evitando o esforço computacional necessário para fazer a validação de todos os pixels da cena através da comparação de seu posicionamento em relação à viewport, otimizando o processo de recorte circular, assim como a detecção e remoção de superfícies escondidas.

Palavras-chave: *Clipping, janela de visualização circular, hashing*

Referências

[1] A. Drozdek. Data Structures and Algorithms in C++. Brooks/Cole Publishing Co., Pacific Grove, CA, USA, 2000.

[2] Q.Wu, X. Huang, and Y. Han. A clipping algorithm for parabola segments against circular windows. Computer & Graphics, (30):540-560, 2006.

[3] M. Zhang and C. L. Sabharwal. An efficient implementation of parametric line and polygon clipping algorithm. In SAC ’02: Proceedings of the 2002 ACM symposium on Applied computing, pages 796-800, New York, NY, USA, 2002.