OPERAÇÕES BOOLEANAS EM POLÍGONOS ARBITRÁRIOS

Aluno: Rafael V. Carvalho
Supervisora: Cristina G. Fernandes
rafaelgtvc@usp.br

Introdução

Operações Booleanas são usualmente estudadas em outras áreas como: lógica, teoria dos conjuntos, álgebra etc. Quando se trata de polígonos essas operações servem, como exemplo, para encontrar: à região em comum entre dois polígono, à região coberta por dois polígonos etc. Nesse sentido, o uso de tais operações é importante para área de computação gráfica, na qual precisa-se determinar quais arestas dos objetos precisam ser renderizadas; e para softwares como CAD e GIS.

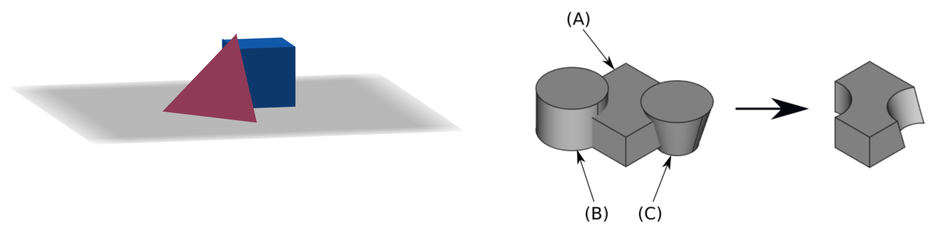

Figure 1: A figura à esquerda mostra dois sólidos, note que uma das arestas do cubo não precisa ser renderizada. A figura à direita mostra um exemplo de uso de tais operações no software CAD

Definições

Antes de explicar o algoritmo, precisa-se introduzir alguns conceitos utilizados por ele e definir o que é um polígono.

Curva poligonal

Segundo [2], uma *curva poligonal* P é definida por uma sequência $(p_1, p_2, \ldots, p_n)$ de pontos no plano chamados de *vértices*. A curva P consiste nos segmentos $\overline{p_i p_{i+1}}$ entre cada par de pontos consecutivos da sequência. Esses segmentos serão chamados de *arestas*.

A curva poligonal P é *aberta* se $p_n \neq p_1$, é *fechada* se $p_n = p_1$ e é *simples* se não se auto-intersecta. Assim uma curva poligonal pode ser de quatro tipos: aberta simples, fechada simples, aberta não-simples e fechada não-simples.

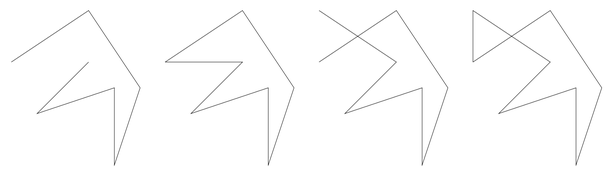

Figure 2: Tipos de curva

Even odd rule e polígonos

Dado um ponto q e uma curva poligonal P, a even odd rule consiste em traçar uma semirreta partindo de q. O número de vezes que essa semirreta se intersecta com P determina se o ponto q está dentro ou fora de P. Se o número de interseções for par, então q está fora, se for ímpar então q está dentro.

Posto isso, um polígono, para o algoritmo estudado, são todas as regiões que tem o numero de interseções ímpar com a semirreta.

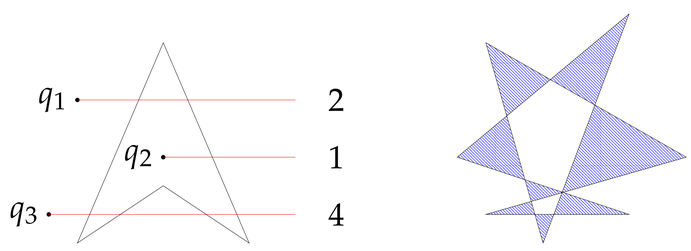

Figure 3: A esquerda tem-se três pontos com o número de vezes que suas semirretas intersectam o polígono. A direita um polígono reconhecido pelo algoritmo

Algoritmo de Greiner-Hormann

O algoritmo de [1] foi proposto em 1989 e é considerado um dos mais elegantes, por conta de sua simplicidade em resolver o problema.

Estrutura de dados

A estrutura de dados utilizada para representar os polígonos é uma lista circular duplamente ligada com os seguintes atributos: vertex, corresponde ao par ordenado (x, y); next, apontado para a próxima célula da lista; prev, apontador para a célula anterior da lista; nextPoly, apontador para outro polígono; intersect, flag que indica se o vértice é de interseção; entry_exit, flag que indica se o vértice é de entrada ou saída; neighbor; apontador para a célula correspondente do outro polígono, caso seja de interseção; alpha; número que indica a localização do vértice em relação à aresta.

Primeira etapa

Essa primeira etapa consiste em encontrar os pontos onde ocorre interseção entre a fronteira dos polígono P e Q. Para isso o algoritmo utiliza uma estratégia de força bruta. Quando um ponto de interseção é encontrado ele é adicionado na lista de P e de Q ordenado pelo atributo alpha. Após isso, a cópia em P aponta para a cópia em Q através do atributo neighbor e vice-versa, a flag intersect é assinalada e o valor alpha é atribuido.

Segunda etapa

Nessa segunda etapa determina-se se os vértices de interseção são de entrada ou de saída. Para isso, executa-se a even odd rule em um dos vértices de P. Após isso, sabe-se se o primeiro ponto está dentro ou fora de Q. Guarda-se essa informação em uma variável e, percorre-se a fronteira de P e a medida que cada vértice de interseção é encontrado, o valor dessa variável é alterado para o oposto e salvo no atributo entry_exit, isto é, se o valor era dentro, então agora passa a ser fora e vice-versa. Esse processo também será aplicado ao polígono Q.

Terceira etapa

Na terceira etapa far-se-á uma filtragem na estrutura de dados a fim de descobrir o polígono resultante da operação entre P e Q. Para isso, faz-se o seguinte para o polígono P: encontre um vértice i de interseção que não foi processado; após encontrar tal vértice, percorre-se a lista de P usando o atributo next se i for de entrada, se i for de saída a lista é percorrida utilizando-se o atributo prev; quando encontrar um próximo vértice de interseção mudamos para a lista de Q através do atributo neighbor; executa-se esse processo até todos os vértices de interseção sejam processados e, a medida que os vértices são visitados, o polígono resultante vai sendo construído.

Casos Degenerados

Os casos degenerados ocorrem quando uma interseção entre duas arestas se da no extremo de um vértice, ou seja, quando o atributo alpha é 1 ou 0. Para tratar esse caso [1] realiza um pequena pertubação aleatória em todos os vértices dos polígonos.

Conclusão

Embora o algoritmo [1] seja relativamente simples, há dois pontos negativos que precisam ser destacados. O primeiro, o tratamento dos casos degenerados pode não ser adequado para quando necessita-se de precisão na operação entre dois polígonos. O segundo, utilizar força bruta para encontrar os pontos de interseção entre a fronteira de dois polígonos pode demorar muito tempo se o número de arestas dos polígonos for grande.

References

[1] G. Greiner and K. Hormann. Efficient clipping of arbitrary polygons. *ACM Transactions on Graphics*, 17:71–83, 1989.

[2] Wikipedia. Polygonal chain — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Polygonal%20chain&oldid=1058902113`, 2021. [Online; accessed 23-May-2022].