CI1338 - Tópicos em Geometria Computacional - Segundo Trabalho

Nome: Isadora Botassari

GRR: 20206872

1. Introdução

Neste trabalho, o problema consiste em encontrar uma triangulação para um dado polígono simples. A entrada consiste no número *n* de vértices do polígono, seguido da descrição de cada vértice do polígono. De acordo com o enunciado, a estrutura de dados que representa a triangulação possui as seguintes características:

"*Cada vértice recebe um índice, de 1 a n, e suas coordenadas são armazenadas em um vetor indexado por estes índices. Cada triângulo recebe um índice, de 1 a m, e um vetor indexado por estes índices deve conter: três campos para os vértices do triângulo, V1, V2, e V3, com os índices dos três vértices; três campos com os índices dos triângulos vizinhos, T1, T2 e T3, de forma que o triângulo Ti seja oposto ao vértice Vi. Os vértices de um triângulo devem estar em ordem horária e o primeiro índice (V1) deve ser o menor. Caso a face externa não seja um triângulo, esta recebe o índice 0.*"

2. Estruturas de Dados

As seguintes estruturas de dados foram montadas para a resolução do problema.

Ponto: estrutura que representa um vértice ou um ponto qualquer. Possui as coordenadas x,y, um valor booleano de convexidade ou não convexidade, e um índice numerado.

Aresta: estrutura que representa uma aresta do polígono, ou então, da triangulação. Armazena os dois Pontos que a definem (início e fim), além dos índices dos triângulos dos quais a aresta participa.

Triângulo: estrutura que representa um triângulo. Possui um índice numerado que o identifica na triangulação, além de três vetores com três posições cada um: o de vértices, o de arestas e o de vizinhos.

3. Algoritmos Utilizados

O algoritmo de triangulação implementado foi o método de corte de orelhas (*ear clipping*). É um algoritmo simples e que possui complexidade $O(n^2)$.

A entrada é lida e, a partir dela, um vetor de vértices é construído. Nessa construção, além das coordenadas, cada vértice recebe um índice (de 1 a *n*) e uma classificação de convexo ou não convexo. A convexidade é obtida a partir do cálculo vetorial entre a aresta formada pelos vértices *n-1* e *n* e pela aresta formada pelo vértices *n* e *n n+1*. Ainda como parte do pré-processamento, para evitar problemas relativos ao sentido dos vértices fornecidos na entrada (horário ou anti-horário), é realizado um processo de complexidade O(n) para padronizar todos os polígonos como horários.

O laço itera pelos vértices do polígono e testa se é possível construir um triângulo com um certo vértice *i* e seus vizinhos (*i-1* e *i+1*). Para isso, dois testes são realizados: o primeiro verifica se nenhum vértice do polígono original estaria dentro desse possível triângulo; o segundo, se o vértice do "meio", ou seja, *i*, é um vértice convexo, ou então, se todos os vértices testados são não convexos. Neste trabalho, foi convencionado que vértices e arestas no sentido horário são convexos.

Em caso positivo nos dois testes, o triângulo é construído e inserido no vetor de triângulos da triangulação. Por fim, o vértice *i* do triângulo formado é removido do vetor de vértices, assim

realizando o “corte de orelha” no polígono. O laço itera por tríades de vértices até o momento que o número restante é menor do que 3. Este processo possui complexidade $O(n^2)$, e gera um vetor com m triângulos.

O processo seguinte é o cálculo dos vizinhos de cada aresta da triangulação. Cada aresta pode fazer parte de um ou dois triângulos; dessa forma, a estrutura de dados possui campos para dois vizinhos. O primeiro é definido na construção do triângulo: se uma aresta *u,v* foi utilizada para construir um certo triângulo, logicamente ele é um dos vizinhos de *u,v*. A função calcularVizinhos realiza uma busca num lista ordenada (critério de ordenação: coordenada x do ponto de início) de arestas com o objetivo de, dada uma aresta *u,v* encontrar a mesma aresta no “sentido oposto”, ou seja, *v,u*. Para cada aresta *u,v*, uma busca é realizada dentro de intervalo de arestas cujo tamanho é variado (mas não corresponde à lista inteira). Esse intervalo é composto pelas arestas cujo ponto final possui a mesma coordenada x que o ponto inicial da aresta *u,v.* Se essa aresta é encontrada, *u,v* não é uma das arestas da borda e, portanto, está na fronteira entre dois triângulos. Essa informação é usada para definir o par de vizinhos de uma aresta.

Por fim, para a formatação da saída, os vértices que formam cada triângulo são ordenados; isto possui complexidade $O(m)$, pois uma ordenação é feita para cada triângulo.