Mecánica Computacional, Volumen XVIII - pp. 509-518
Compilado por: E. Dari, C. Padra y R. Saliba
San Carlos de Bariloche, noviembre de 1997

UM MODELO PARA GERAÇÃO DE GRADES TRIÂNGULARES

Conrado S. S. Zenun
Escola Federal de Engenharia de Itajubá
Departamento de Matemática e Computação
Av. BPS, 1303 - Itajubá MG - 37500-000

Márcio L. X. dos Santos
Instituto Tecnológico de Aeronáutica
Departamento de Computação Científica
Instituto Tecnológico de Aeronáutica
CTA - São José dos Campos SP - 12220-900
e
Universidade de Taubaté
Departamento de Informática
Taubaté - SP

RESUMO

A discretização por triângulos ou triangulação, é o processo de seleção de triplas que formam facetas triangulares definidas em estruturas computacionais eficientes. Foram estudados vários algoritmos de triangulação que geram triângulos de Delaunay. Desenvolveu-se um modelo de triangulação em três etapas, na primeira etapa faz-se o pré-processamento, na segunda etapa gera-se uma grade triangular inicial e na terceira etapa realiza-se a otimização da grade inicial, pelo método da maximização do menor ângulo interno. O algoritmo desenvolvido usa abordagens conhecidas, no entanto, oferece de forma inovadora a possibilidade de tratar simultaneamente conjuntos 2D convexos ou não, e superfícies suaves 3D, para qualquer distribuição de pontos . Superfícies 2D , convexas e não convexas e superfícies 3D convexas foram modeladas e alguns resultados são apresentados. O modelo de facetamento da superfície gerada pode ser utilizado em técnicas de iluminação e sombreamento para visualização ou para o cálculo das propriedades relevantes de superfícies arbitrárias , bem como, em aplicações de elementos finitos.

ABSTRACT

Triangulation of a domain is a discretization process in which a selection of triples pertaining to one of the various possible collections of three distinct elements of a set within the domain is found such that every triple can be used as a triangular facet to model that domain. The ideal triangulation process should render a domain decomposition into a triangular grid which should be simple to generate, comfortable to use and computationally efficient. To select a triangular grid exhibiting these properties is far from being an easy task. In short, what is sought of in this problem is a triangulation schemme which may lead to Delaunay triangles. In the present paper a computerized triangulation method accomplished through three refinement phases is resented. During the first phase the set is pre-processed to organize the candidate triples. In the second phase an initial temptative triangular grid is formed. Finally, this temptative grid is automatically refined by optimizing the triangular facets so that each triangle exhibits the greatest minimum internal angle. This innovative algorithm, although based on well known ideas, can be used advantegeously to discretize 2D convex and non-convex sets and also 3D smooth surfaces for any given set of discrete surface.

INTRODUÇÃO

A evolução histórica dos métodos de triangulação acompanha a evolução dos computadores e, por consequência, a evolução das suas aplicações. Inventada por Snellius por volta de 1615 [1], foi utilizada até a década de 70 por topógrafos em problemas para representar dados de terrenos.

Com a evolução da computação, vários algoritmos para a discretização de superfícies foram propostos. Os primeiros algoritmos eram dedicados a aplicações do método de elementos finitos. Muitos desses algoritmos enfatizam aspectos teóricos somente, o que implica que a implementação não é uma tarefa trivial, conforme concluem [2, 3]. Os primeiros algoritmos voltados para o estudo de elementos finitos [4], consistiam em subdividir um dado domínio, com pontos espalhados regularmente, formando triângulos. Tais métodos eram limitados a domínios regulares e são chamados de construtores ou geradores de grades triangulares. Hoje os principais métodos podem ser classificados em dois tipos: algoritmos incrementais, [5, 6, 7, 8, 9] e algoritmos baseados na metodologia "dividir para conquistar", [9, 10]. Ambos os métodos geram uma triangulação de Delaunay, [11].

Algoritmos incrementais iniciam a construção da grade triangular a partir de um ponto qualquer do interior do conjunto que será triangularizado, ou de um ponto extremo desse conjunto. A partir desse ponto, juntam-se seus pontos vizinhos mais próximos, um por vez, construindo assim triângulos sobre todo o domínio.

Algoritmos que dividem para conquistar, dividem o conjunto de pontos em subconjuntos de cardinalidade igual até que subconjuntos elementares sejam obtidos. A triangulação então é iniciada unindo-se os pares de subconjuntos.

Os algoritmos que obtêm uma triangulação de Delaunay são ainda classificados pela maneira como produzem a triangulação final. São então chamados de algoritmos de um e dois passos.

A triangulação em um passo [8, 9], constrói cada triângulo como triângulo de Delaunay e, no final, tem-se de uma vez uma triangulação de Delaunay. Nos algoritmos de dois passos [12, 2,10], inicialmente constrói-se uma triangulação arbitrária que será, no segundo passo, otimizada aplicando-se iterativamente critérios de otimização.

Outra abordagem utilizada consiste em construir um diagrama de Voronoi de um dado conjunto de pontos e então computar seu grafo dual, que é a triangulação. Esta abordagem, entretanto, requer inicialmente a construção do diagrama de Voronoi, o que exige grandes estruturas de dados que armazenam informações desnecessárias para a triangulação, [10].
Algoritmos mais recentes, contrastando com os modelos citados anteriormente (algoritmos estáticos), são chamados de algoritmos dinâmicos, [7]. Esses algoritmos permitem atualizações da triangulação com inclusão de pontos em determinadas regiões.

Apesar da grande quantidade de trabalhos publicados nessa área, ainda se buscam novos algoritmos e estruturas de dados que possam apresentar erros com uma faixa estreita de tolerância, estruturas mais simples, tratamento de casos degenerados etc.

O recente trabalho de Fang e Piegl [2], aplicado somente a superfícies 2D traz uma nova proposta, que pressupõe eliminar a necessidade de um pré-ordenamento dos pontos além de tratar pontos coincidentes e colineares. Entretanto, faz-se necessário estudar mais o método proposto.

Observa-se que mesmo aspectos aparentemente elementares, como o processo de otimização de uma triangulação, ainda são motivos de pesquisa, porque incluem questões cuja solução ótima continua em aberto.

DESCRIÇÃO GERAL DO TRABALHO

A literatura a respeito dos algoritmos de triangulação é extensa, sendo que os aspectos teóricos são mais enfatizados tais como complexidade e eficiência.

Dada a vasta área de aplicação, esses algoritmos são geralmente dirigidos aos casos específicos. As questões chamadas de situações degeneradas, como pontos coincidentes, pontos colineares e cocirculares, nem sempre são abordadas, dificultando ou restringindo a implementação dos algoritmos. Outra questão importante diz respeito à natureza dos pontos do domínio em questão. Eles podem ser do tipo $P_i = f(x_i, y_i)$ ou $Z_i = f(x_i, y_i)$ ou $P_i = f(x_i, y_i, z_i)$, $i = 1, \ldots N$, onde N é o número total de pontos do conjunto.

Cada conjunto pode ainda ser convexo ou não convexo, com distribuição regular ou aleatória dos pontos. Discutir com exaustão, teoria, algoritmos, estruturas de dados sobre triangulação, é tarefa difícil senão impossível.

A seguir, apresentamos um algoritmo de maneira detalhada tal que possa ser entendido e implementado.

Procuramos com isto, cobrir uma lacuna existente na literatura, ou seja, a dificuldade de traduzir propostas teóricas para uma implementação eficiente.

O algoritmo desenvolvido usa abordagens conhecidas; no entanto, oferece de forma inovadora a possibilidade de tratar simultaneamente conjuntos 2D convexos ou não, e superfícies suaves 3D, para qualquer distribuição de pontos. As questões degeneradas são tratadas de maneira simples, porém com bons resultados [13].

O ALGORITMO DE FACETAMENTO TRIANGULAR

O algoritmo proposto por Choi et alli [12], foi utilizado como modelo básico para o sistema desenvolvido, e será discutido em detalhes com modificações e interpretações diferentes do texto original.

Estas mudanças fizeram-se necessárias para o perfeito entendimento e implementação do algoritmo.
O algoritmo está dividido nas seguintes partes: pré-processamento, triangulação inicial, triangulação principal, triangulação para superfícies planas não convexas e otimização.

O PRÉ-PROCESSAMENTO

1° Passo:
Defina a superfície S, sendo P_j o conjunto de pontos pertencentes a S.

2° Passo:
Encontre um ponto C que será o ápice de um cone convexo de geratriz $\vec{u}$ que inscreverá S, veja figura 1.1.

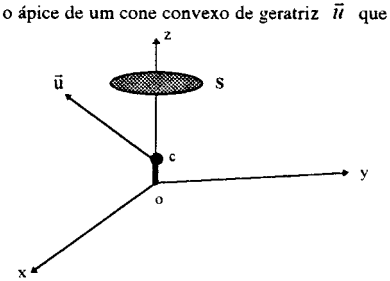


Fig. 1.1. Definição do cone convexo.

3° Passo:
Mude a origem dos pontos $\{P_j\}$, de modo que C torne-se a origem. Fazendo a transformação obtemos:

$$V_j = P_j - C$$

sendo V_j o conjunto de pontos de S na nova origem.

4° Passo:
Ordene os pontos $\{ \vec{V}_j \}$ em ordem crescente em relação ao ângulo entre os dois vetores $\vec{u}$ e $\vec{V}_j$.

TRIANGULAÇÃO INICIAL

1° Passo:

Armazene os dois primeiros vértices da lista de vértices ordenados, na TBL (veja figura 1.2c).

2° Passo:

Obtenha o terceiro vértice V.

3° Passo:

Enquanto ($\vec{V}$ está sobre $\overrightarrow{V1V2}$) faça:

- se ($\vec{V}$ está mais próximo do vértice inicial) então
  insira-o no início da TBL; (figura 1.2d)
- se ($\vec{V}$ está mais próximo do vértice final) então
  insira-o no fim da TBL; (figura 1.2e)
- obtenha o novo vértice;

fim enquanto;

4° Passo:

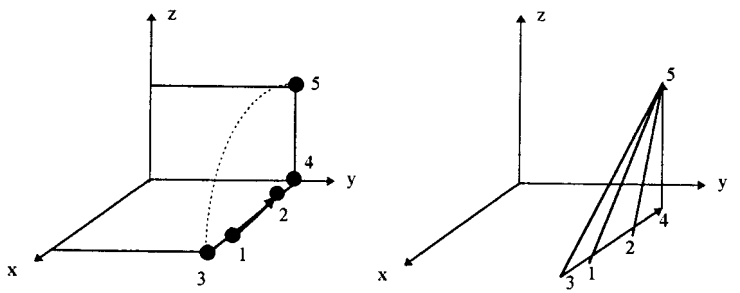
Se (V estiver a direita de $\overrightarrow{V1V2}$) então
troque a direção de TBL;

5° Passo:

Insira V no início da TBL e armazene os números dos triângulos; (veja figura 1.2f);

6° Passo:

Construa a LTL para a grade inicial; (veja figura 1.2b);



a) Exemplo didático

b) Grade inicial

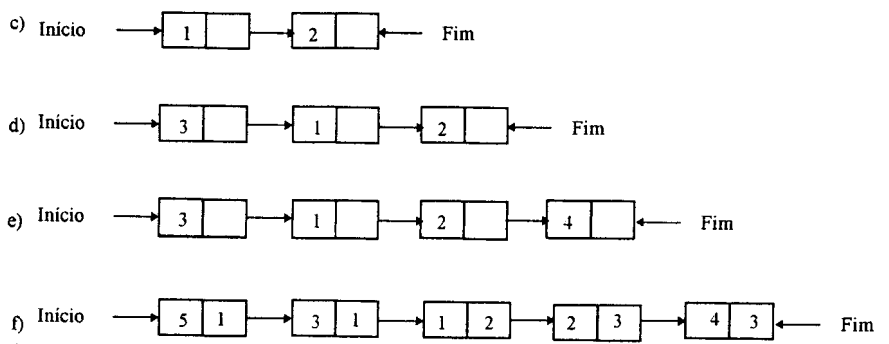


Fig. 1.2 (a-f) - Construção da triangulação inicial

TRIANGULAÇÃO PRINCIPAL

Repita

1° Passo:
Obtenha o próximo vértice da lista de vértices e coloque-o no novo nó NN.

2° Passo:
Encontre os nós mais visíveis à esquerda (LVN) e à direita (RVN) dos nós da TBL, (veja figura g);

3° Passo:
Construa um triângulo que consiste dos nós LVN, NN e o nó que segue LVN na TBL, (veja figuras h e j);

4° Passo:
Insira NN depois de LVN na TBL, (veja figura h);

5° Passo:
Continue construindo triângulos até que a TBL torne-se convexa, eliminando da TBL o nó que segue LVN, (veja figuras i, l e m);

até (lista de vértices vazia);

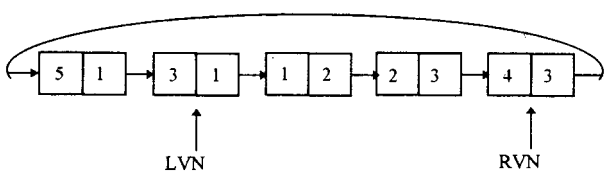


Figura g

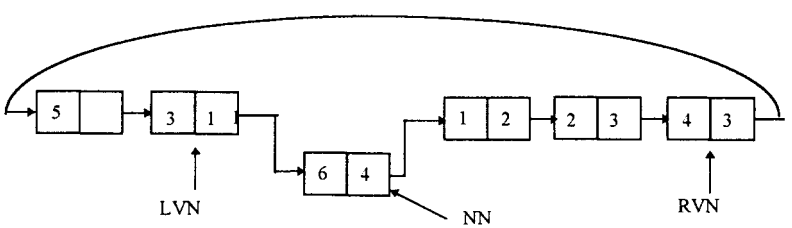


Figura h

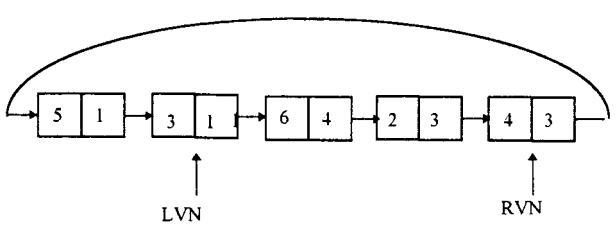


Figura i

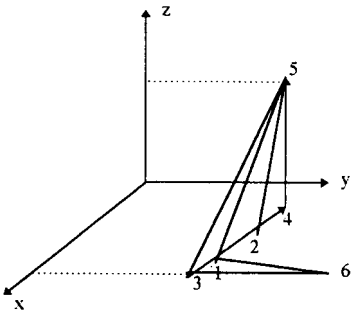


Figura j

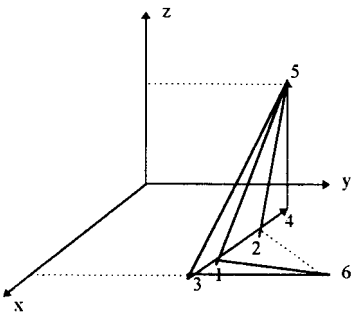


Figura l

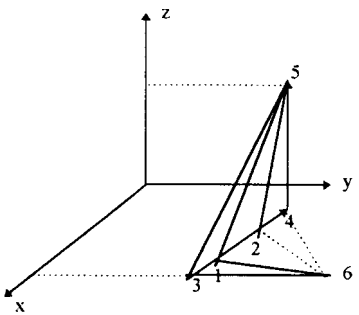


Figura m

Fig. 1.3 (g-m) - Construção da grade triangular principal.

TRIANGULAÇÃO DE SUPERFÍCIES PLANAS NÃO CONVEXAS

Superfícies que apresentam "buracos" podem ser triangularizadas adotando-se o seguinte algoritmo:
1° Passo:
Marcar pontos (ou vértices) que pertencem à borda interna ou externa da região que apresenta um "buraco".
Veja figura 1.4.

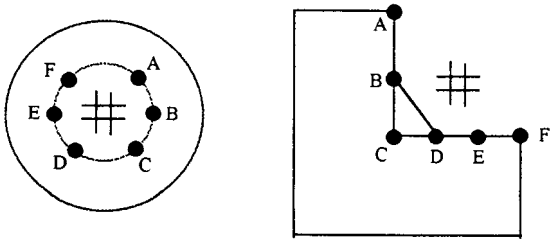


Fig. 1.4 - Exemplos de superfícies não convexas.

Os pontos A, B, C, D, E, e F, nas figuras acima serão marcados.

2° Passo:
Identificar as bordas, criadas pela triangulação principal, que pertencem a fronteira marcada.

3° Passo:
Identificar e remover os triângulos que possuem uma dessas bordas.

OTIMIZAÇÃO DA GRADE TRIANGULAR

OTIMIZAÇÃO 2D

O primeiro processo de otimização estudado consiste em encontrar um quadrilátero estritamente convexo e aplicar sobre este quadrilátero o método da maximização do menor ângulo interno. Em linguagem algorítmica os dois passos acima podem ser descritos como:
Repita (para cada borda)

1° Passo: Recuperação dos triângulos à esquerda e à direita da borda que faz interseção com os dois triângulos do quadrilátero, ou seja, a diagonal do quadrilátero, a partir das informações armazenadas na lista de bordas, ELIST.

2° Passo: Recuperação dos vértices de cada triângulo também a partir da ELIST.

3° Passo: Encontrar os ângulos internos ao quadrilátero.

4° Passo: Trocar a diagonal do quadrilátero.

5° Passo: Encontrar os novos ângulos internos ao quadrilátero.

6° Passo: Escolher a diagonal que provoca ângulos internos maiores.

até que (não se verifique a necessidade de se trocar diagonais).

OTIMIZAÇÃO 3D

O segundo processo de otimização estudado consiste em encontrar um quadrilátero estritamente convexo e aplicar sobre este quadrilátero o método da "suavidade" que consiste em encontrar qual diagonal que gera superfícies mais suaves. Algoritmicamente teremos.
Repita (para cada borda)

1° Passo: Recuperação dos triângulos à esquerda e à direita da borda que faz interseção com os dois triângulos do quadrilátero, ou seja, a diagonal do quadrilátero, a partir das informações armazenadas na lista de bordas, ELIST.

2° Passo: Recuperação dos vértices de cada triângulo também a partir da ELIST.

3° Passo: Encontrar a normal à face de cada triângulo.

4° Passo: Encontrar o ângulo entre as normais.

5° Passo: Trocar as diagonais.

6° Passo: Encontrar as novas normais às faces de cada triângulo.

7° Passo: Encontrar o novo ângulo entre as normais.

8° Passo: Admitir a diagonal que gerou ângulo maior.

até que (não se verifique a necessidade de se trocar diagonais).

RESULTADOS

A seguir são apresentados algums resultados gerados pelo algoritmo descrito acima. A figura do quadro esquerdo mostra a grade não otimizada e a do lado direito a grade otimizada.

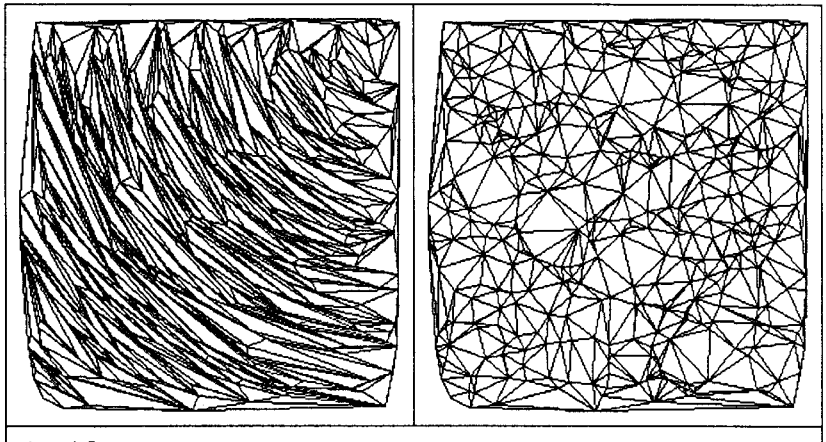


Fig. - 1.5 Superficie PLANA
Num. de Pontos = 400
Num. de Triangulos = 783

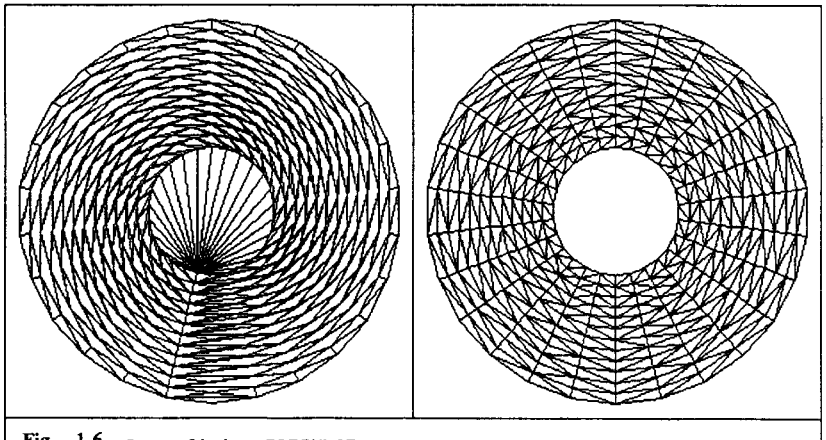


Fig. - 1.6 Superficie CIRCULAR
Num. de Pontos = 338
Num. de Triangulos = 648

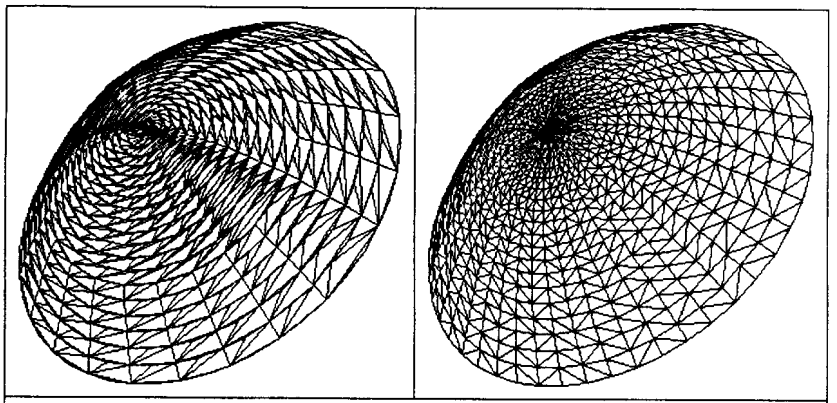


Fig. - 1.7 Superficie PARABOLICA
Num. de Pontos = 841
Num. de Triangulos = 1638

BIBLIOGRAFIA

[1] AGISHTEIN, M.E. and MIGDAL, A.A. - Smooth Surface Reconstruction from Scattered Data Points - Computer & Graphics, vol. 15, n° 1, p. 29-39, 1991.

[2] FANG, T.P. and PIEGL, L.A. - Delaunay Triangulation Using a Uniform Grid - IEEE Computer Graphics & Applications. p. 36-47, May 1993.

[3] DAY, A.M. - The Implementation of an Algorithm to Find the Convex Hull of a Set of Three-Dimensional Points - ACM Transaction on Graphics - Vol. 9, n° 1, p. 105-131, 1990.

[4] RIVARA, M.C. - Algorithms for Refining Triangular Grids Suitable for Adaptive and Multigrid Techniques - Int. Journal for Numerical Methods in Engineering - Vol. 20, p. 745-756, 1984.

[5] BARNHILL, R.E. - Representation and Approximation of Surfaces - Mathematical Software III, Academic Press - New York, USA - 1977.

[6] BOWYER, A. - Computing Dirichlet Tessellations - The Computer Journal - Vol. 24, n° 2, p. 162-166, 1981.

[7]GUIBAS, L. and STOLFI, J. - Primitives for the Manipulation of General Subdivisions and the Computation of Voronoi Diagrams - ACM Transactions on Graphics - vol. 4 - n° 2, p. 75-123, 1985.

[8] LAWSON, C.L. - Softwares for C^1 Surface Interpolation - Mathematical Software III, Academic Press - New York, USA, 1977

[9] LEE, D.T. and SCHACHTER, B.J. - Two Algorithms for Constructing a Delaunay Triangulation - International Journal of Computer and Information Sciences, vol. 9, n° 3, p. 219-241, 1980.

[10]PREPARATA, F.P. and SHAMOS, M.I. - Computational Geometry (An Introduction) - Springer Verlag, 1985.

[11]DE FLORIANI, L. - Surface Representations Based on Triangular Grids - The Visual Computer, n° 3, p. 27-50,1987.

[12]CHOI, B.K. et alli - Triangulations of Scattered Data in 3D Space - Computer Aided Design - vol. 20, n° 5, p. 239-248, June 1988.

[13]A cópia completa deste trabalho pode ser solicitada aos autores.