MINISTÉRIO DA CIÊNCIA E TECNOLOGIA

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

GERAÇÃO DE MALHAS TRIANGULARES NÃO-ESTRUTURADAS

José Jean Peixoto Negrão

Dissertação de Mestrado em Computação Aplicada, sob orientação do Dr. Jerônimo dos Santos Travelho

INPE

São José dos Campos

Maio de 1997

Aprovada pela Banca Examinadora em cumprimento a requisito exigido para a obtenção do Título de Mestre em Computação Aplicada

Dr. Haroldo Fraga de Campos Velho — Presidente

Dr. Jeronimo dos Santos Travelho — Orientador

Dr. Luiz Antonio Nogueira Lorena — Membro da Banca

Dr. João Luiz Filgueiras de Azevedo — Membro da Banca - Convidado

Candidato: José Jean Peixoto Negrão

São José dos Campos, 10 de marco de 1997

AGRADECIMENTOS

Ao INPE pela disponibilidade de equipamentos. Ao CNPq pelo suporte financeiro. Ao Dr. Jerônimo do Santos Travelho, pela valiosa orientação, aos membros da banca pela atenção e compreensão. Ao Dr. Horácio Hideki Yanasse e aos meus colegas do LAC pelo incentivo e apoio, e finalmente aos meus pais que foram muito importantes para a realização deste trabalho.

RESUMO

Nesta dissertação é apresentado um método de geração de malhas triangulares não-estruturados, especificamente um método tipo avanço da frente modificado. Este técnica possui três estágios: inicialmente é gerada uma malha, que em seguida é suavizada, e depois pode ser refinada e "desrefinada". O método apresentado foi desenvolvido voltado para aplicações em problemas *STIFF* e métodos *Multigrid*.

ABSTRACT

This dissertation presents a unstructured triangular grid generation method. The method is a modified advanced front kind. This technique consists of three stages: initially, triangular grid is generated, after this step the grid is smoothed, and finally it can be refined or unrefined. This method was developped for the use of numerical methods applied to STIFF problems and Multigrid methods.

SUMÁRIO

Lista de figuras

Lista de Símbolos

$O()$	Ordem de Complexidade de um algoritmo
$\boldsymbol{N}$	Número de iterações do algoritmo
n	Número de novos segmentos
ρ	Parâmetro de comprimento máximo permitido para os sub-segmentos gerados ao se refinar um arco ou segmento de reta
δ	Variável que específica a necessidade (valor igual á 1) ou não (valor igual á 2) de se discretizar um arco ou segmento de reta
x	Valor (posição) no eixo das coordenadas
y	Valor (posição) no eixo das abcissas
ϕ	Ângulo equivalente ao comprimento do arco
$\boldsymbol{n}$	Vetor normal ao seguimento de reta que é dicretizado
$p1$	Ponto da origem do segmento é gerado durante a discretização do contorno
$p2$	Ponto da extremidade do segmento é gerado durante a discretização do contorno
φ	Ângulo cuja a secante tem um comprimento menor ou igual a ρ
θ	Ângulo interno ao fronte atual
β	Ângulo crítico, ou seja, ângulo interno máximo permitido para um triângulo
ψ	Ângulo que será incrementado durante a discretização entre dois segmentos de retas da frente

C_T	Ponto central de cada triângulo que tem como vértice o ponto C_P
C_P	Ponto central ao polígono que será suavizado
$C_P{}^{I}$	Centro exato do polígono depois de suavizado
$C_P{}^{II}$	Centro deslocado ao utilizar um fator de sobre-relaxação

CAPITULO 1

INTRODUÇÃO

Neste trabalho é apresentado um Gerador de Malhas Triangulares Não-Estruturadas para aplicações em Fenômenos de Transportes Computacional, especificamente problemas *Stiff* e métodos tipo Multigrid que estão sendo analisados no LAC/INPE. Nos problemas *Stiff* assim como nos métodos Multigrid, muitas vezes é necessário que se refine e "desrefine" gradualmente a malha em diferentes regiões do domínio, isto a cada passo de iteração do algoritmo do usuário.

O processo de geração de malhas bidimensionais consiste, basicamente, em discretizar o domínio de um modelo físico por meio de pontos e segmentos de retas. Os segmentos de retas delimitam pequenos volumes de controle em duas dimensões, tais como retângulos, paralelogramos, triângulos ou polígonos. Estes pequenos volumes de controle quando somados aproximam-se em forma e dimensão ao modelo físico.

Um dos propósitos de se discretizar domínios físicos através de pequenos volumes de controle advém das facilidades de se trabalhar sempre em sub-domínios com formas simples, tais como os citados acima. Estes sub-domínios podem iterar entre si, permitindo assim a propagação de informações contidas nos mesmos e em seus vértices e arestas. A propriedade de iteração destes sub-domínios é explorada pelos Métodos de Diferenças Finitas, Elementos Finitos e Volumes Finitos. Estes Métodos tem por finalidade resolver de forma discreta equações da Mecânica dos Fluídos sobre os domínios já discretizados.

Poderíamos classificar as diferentes técnicas de geração de malhas conhecidas de várias maneiras. Entretanto, neste trabalho elas são colocadas em três categorias ou tipos distintos: A primeira categoria seria a das Malhas Retangulares Estruturadas, onde os volumes de controle são retângulos e seus vértices são pontos internos ao domínio. Estes vértices se relacionam com seus vizinhos de uma forma estruturada, ou seja, para cada vértice interno ao domínio, existem sempre quatro outros vizinhos, de maneira que se pode utilizar sempre uma relação como mostrado na **fig1.1**. Estas malhas podem ser regulares, ou seja, para cada vértice interno ao domínio, os vértices vizinhos estão a uma distância constante uns dos outros, em cada uma das direções (**i** ,,**j**), ou podem ser irregulares, onde a regra citada não é verdadeira. Devido a forma retangular dos volumes de controle, este tipo de malha não se adapta a domínios com contornos físicos não retangulares, tais como a asa de um avião por exemplo. Entretanto, ainda é possível aplicá-los a estes domínios, desprezando-se detalhes destes, ou interpolando-se sobre a região não retangular.

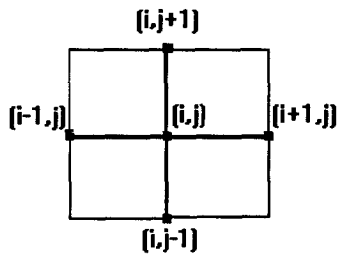


fig1.1 *Relação de vizinhança de vértices em malhas retangulares estruturadas.*

A segunda categoria seria a das Malhas Adaptativas/Coincidente ao Contorno, ou em Coordenadas Generalizadas. Neste tipo, os volumes de controle podem admitir formas diversas, de acordo com as curvas do contorno do domínio em questão. Aqui apresentamos um exemplo simples **fig1.2**, onde os volumes de controle são paralelogramos e seus vértices são os pontos internos ao domínio. Neste caso, estes vértices se relacionam com seus

vizinhos da mesma forma que na categoria das Malhas Retangulares. Estas malhas também podem ser regulares ou não. Os métodos que geram estes tipo de malha, utilizam-se de pares de curvas coincidentes com o contorno do domínio. Estas curvas são interpoladas para gerar os pontos internos ao domínio, e os pontos gerados compõem uma malha estruturada sobre o mesmo. A possibilidade de se poder moldar o formato para os volumes de controle de acordo com o as curvas que descrevem este contorno, permite que estas malha se adaptem bem a domínios físicos não retangulares. Entretanto, para domínios com contornos muito irregulares fica muito difícil encontrar curvas que coincidam exatamente com o contorno.

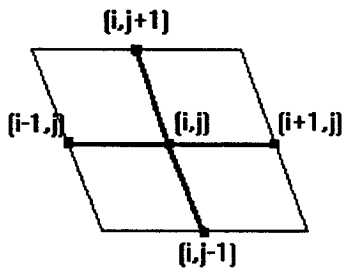


fig.1.2 Relação de vizinhança de vértices em malhas Adaptativas

A terceira categoria é a das Malhas Triangulares, onde os volumes de controle são triângulos. O formato triangular dos volumes de controle, permite que estes vértices se relacionem com seus vizinhos de uma forma estruturada **fig.1.3.a** ou não **fig.1.3.b**. Além disso, é possível se adaptar a malha discretamente a domínios físicos não retangulares e bem irregulares. É possível ainda, determinar exatamente onde se deve concentrar pontos dentro do domínio. Por estas razões, optamos por utilizar neste trabalho malhas triangulares não estruturadas.

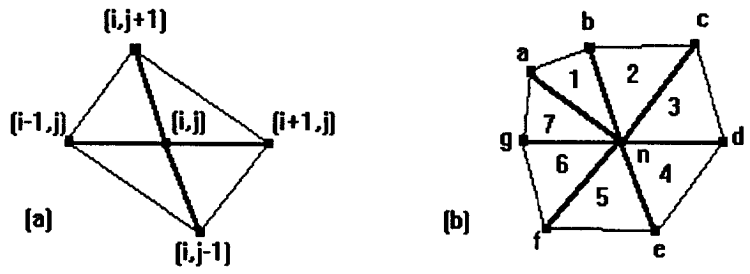


fig.1 Relação de vizinhança de vértices em malhas triangulares

O processo de geração de malhas triangulares não-estruturadas consiste em pegar um contorno de um modelo físico e discretiza-lo, inserindo pontos e gerando triângulos com estes pontos. A concentração de pontos em regiões determinadas é algumas vezes feita durante o processo de geração da malha. Desta forma, se fosse necessário retirar ou inserir novos pontos na malha já gerada, teria que se utilizar um processo de interpolação entre os pontos em torno do local onde se deseja inserir o novo ponto, ou em torno do ponto que se pretende retirar.

O processo mencionado acima, é muito custoso computacionalmente quando aplicado ao tipo de problema que este trabalho se propõe a discretizar, isto devido as inúmeras interpolações necessárias neste caso. Por essa razão, optou-se por utilizar um método de geração de malhas alternativo. Este método alternativo, propõe gerar uma malha inicial espaçada e o mais regular possível, e depois refina-la nas regiões momentaneamente interessantes, e desrefina-la (neste caso, recuperar as células inicialmente refinadas) nas regiões que momentaneamente não são mais interessantes. A recuperação das células já refinadas, pode ser feita armazenando-se as mesmas em um arquivo de dados.

O problema que inicialmente motivou desenvolver este gerador de malhas, foi um problema do tipo "*Stiff*" [1]. Neste modelo, existe uma interface de solidificação com uma forma extremamente irregular que, à medida em que

o material vai se cristalizando, muda constantemente dentro do domínio, tanto na forma como na posição. Assim, a concentração de pontos deve ser feita numa região diferente a cada passo de iteração. Além disso, temos na interface de solidificação, uma variação muito rápida das propriedades estudadas, o que implica em gerar uma malha extremamente adensada em diferentes lugares a cada momento.

As equações de Difusão que descrevem este problema são discretizadas pelo método de Volumes Finitos. Neste caso a variação no tamanho das células não pode ser extremamente rápida, porque neste método os fluxos nas paredes são estimados utilizando-se os valores médios do centro das células vizinhas. Assim, se uma célula é muito menor do que sua vizinha imediata, a interface entre elas estará muito mais próxima do centro da menor célula do que da maior, desta forma o valor estimado estaria muito mais próximo do valor no centro da menor célula do que do valor real na interface. Logo, é necessário gerar uma malha adensada em regiões específicas, mas a passagem da região adensada para a região espaçada deve ser suave. Contudo, a interface de transição nos problemas de difusão dominante não precisa ser tão suave como nos problemas de convecção dominante.

Uma explanação sobre alguns algoritmos de geração de malhas triangulares não-estruturadas é feita por Mavriplis em 1992 [5]. Neste trabalho o autor apresenta seu algoritmo de Avanço da Frente com critério de Triangulação Delaunay, e compara o mesmo com alguns outros algoritmos de triangulação tipo Advancing Front, triangulação Delaunay e outros algoritmos que combinam características e propriedades de ambos os tipos. Desta forma, cita outros autores que desenvolvem ou já desenvolveram trabalhos pertinentes ao tema desta dissertação.

Em 1983 Kirkpatrick [2] apresentou um algoritmo prático para triangulação. Um segundo algoritmo foi apresentado por Edelsbrunner-Guibas-Stolfi em 1986 [3]. Ambos os algoritmos e outros que surgiram posteriormente

são explanados no livro *Computational Geometry An Introduction*, de Franco P. Preparata e Michael Ian Shamos em 1988 [4], onde se fala sobre geometria computacional, e se faz uma boa introdução em geração de malhas triangulares não-estruturadas e seu dual, o diagrama de Voronoi.

Burrough em 1986 [6], apresenta um método que utiliza um grupo de pontos espaçados irregularmente como base para gerar uma triangulação. Em seguida, Chen em 1987 [7], apresenta um método que representa uma superfície através de faces de triângulos. Por conseguinte MacCullagh em 1988 [8] gera triangulações sobre um dado grupo de pontos dentro do domínio, respeitando linhas especiais que não devem ser cruzadas pelas arestas dos triângulos. Felgueiras em 1995 [9] apresenta um algoritmo para triangulação Delaunay com linhas de restrição. Este algoritmo respeita não só o critério de triangulação Delaunay, bem como as linhas especiais da mesma forma que o algoritmo de MacCullagh.

Riemslagh em 1994 [10] apresenta um método de relaxação de múltiplos estágios de Jacobi em métodos Multigrid para equações estacionárias de Euler, em uma malha não-estruturada bidimensional. O autor também faz referência a alguns algoritmos de triangulação Delaunay, e descreve especificamente o algoritmo de Tanemura [11].

Briceño em 1994 [12] utilizou um método de refinamento em malhas adaptativas para os métodos de elementos finitos para o problema de Poisson em uma região bidimensional. Nestes problemas, de transferência de calor em placas planas, a temperatura dos lados da placa são conhecidas. A modelagem matemática do problema é feita por meio de um funcional y, e se aplica o refinamento adaptativo segundo a variação deste funcional e segundo a direção do gradiente.

Em 1996, Azevedo e Korzenowski [13] utilizam refinamento adaptativo em malhas não-estruturadas para solucionar escoamentos hipersônicos. Eles apresentam uma comparação entre as várias soluções encontradas para

simulação de escoamento hipersônico numa entrada de ar. Os resultados obtidos indicam que é possível se obter boas soluções para o cálculo de escoamentos utilizando o procedimento de refinamento adaptativo. Também em 1996 [14], eles analisaram escoamentos hipersônicos utilizando malhas não-estruturadas. Neste último, é feita uma avaliação crítica de três esquemas diferentes de discretização espacial para cálculo de escoamentos hipersônicos. Ainda em 1996, Dourado e Azevedo[15] fizeram uma simulação de escoamentos com malhas não-estruturadas sobre configurações automotivas básicas. Neste trabalho é apresentado um método numérico para a solução das equações de Navier-Stokes bidimensionais com média de Reynolds, para todos os regimes de velocidade, em malhas não-estruturadas.

Mathur, Murthy e Choudhury também em 1996 [16] apresentaram um esquema de volumes finitos para escoamento de fluidos reativos. Neste, são utilizadas malhas adaptativas não-estruturadas com uma formulação de célula centrada.

Além destes, vários outros exemplos de aplicações podem ser citados, e diferentes tipos de discretizações podem ser feitas para cada um dos exemplos apresentados. Entretanto, cada modelo em particular deve ter mais afinidade com certos tipos específicos de modelagem e discretização, em detrimento de outros.

No *capítulo 2*, serão apresentados dois algoritmos para gerar malhas triangulares não estruturadas e irregulares, um baseado na triangulação Delaunay e outro tipo Avanço da Frente. Em seguida no *capítulo 3*, são mostradas os três algoritmos utilizados na técnica de geração de malhas propostas. No *capítulo 4*, são apresentados resultados e conclusões. E, finalmente no *capitulo 5*, serão apresentadas propostas para trabalhos futuros.

CAPITULO 2

ALGORITMOS DE TRIANGULAÇÃO

2.1. ALGORITMO DE TRIANGULAÇÃO DELAUNAY.

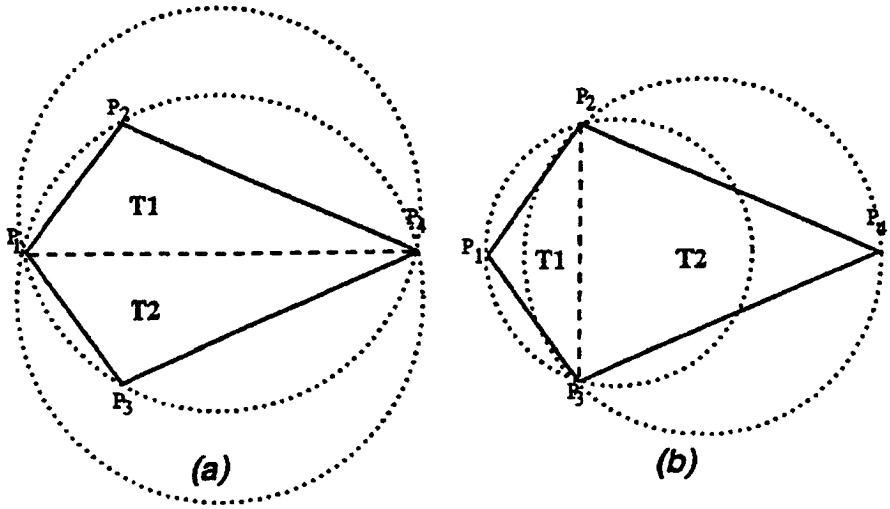


fig.2.1.1 Aplicação do critério do circuncírculo vazio em um par de células

Uma triangulação Delaunay de um dado grupo de pontos, apresenta uma classe de propriedades bem definidas. Uma destas é a propriedade do circuncírculo vazio, **fig.2.1.1,** que especifica que nenhum triângulo na triangulação Delaunay pode conter um outro ponto na sua formação de vértices dentro deste circuncírculo. Desta forma, parte-se de uma configuração com triângulos não Delaunay, **fig.2.1.1.a**, e, após a aplicação de um algoritmo de troca de diagonais, tem-se na **fig.2.1.1.b**, uma configuração com triângulos Delaunay. Esta técnica, mostrada na **fig.2.1.2**, foi utilizada por Lawson no seu algoritmo de geração de malhas triangulares [17].

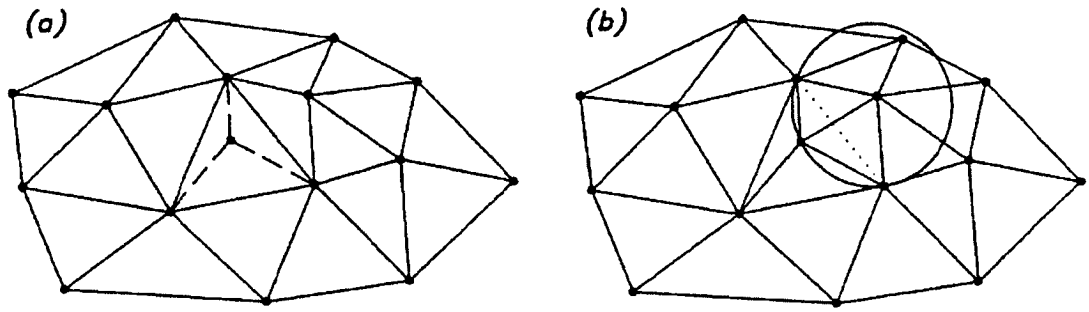


fig.2.1.2. Aplicação do algoritmo de troca de diagonais de Lawson.

Um outro algoritmo que utiliza como base o critério de triângulação Delaunay é o algoritmo de Bowyer/Watson [5]. Neste algoritmo, dada uma triangulação inicial, um novo ponto pode ser inserido na triangulação, pela ação de primeiro localizar e eliminar todos os triângulos existentes cujos circuncírculos contenham o recente ponto inserido. A nova triangulação é formada ligando-se o novo ponto até toda fronteira de vértices da cavidade, criada pela prévia remoção dos triângulos intersectados, assim como é mostrado na **fig. 2.1.3.**

O pior caso de complexidade deste algoritmo$O(N^2)$ [5], acontece quando, a cada nova inserção de pontos, existem triângulos circuncírculos intersectados. Porém, $O(N^2)$ é um comportamento que representa um caso patológico, e em geral aplicações de geração de malhas que empregam estes algoritmos apresentam-se fechados a $O(N)$ [5], ou seja, complexidade linear. De fato, tem-se mostrado que $O(N \log N)$ [5] sempre pode ser alcançada se a ordem na qual os pontos são inseridos for modificada. Para domínios não convexos, a integridade das fronteiras não está garantida por tal técnica. Isto é geralmente remediado, pelo aumento do número de pontos aplicados na fronteira, e pela execução de uma fase de retirar bordas trocadas, assim como uma operação de pós-processamento para recuperar as bordas da fronteira.

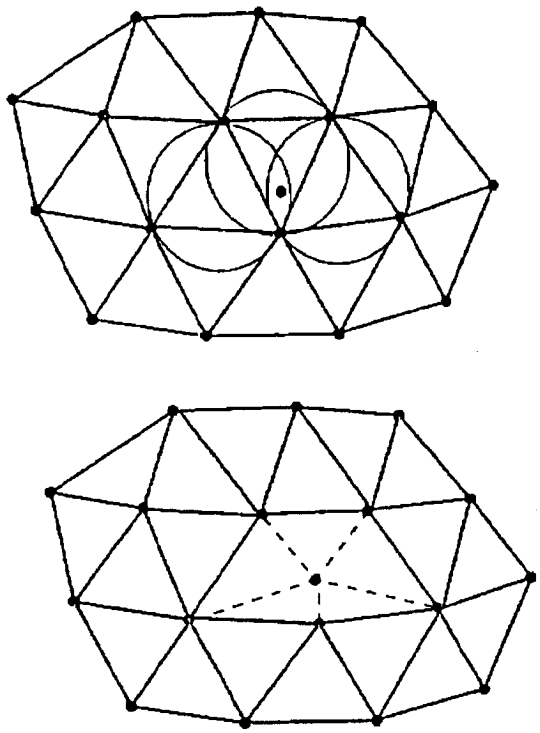


fig. 2.1.3 Ilustração do algoritmo de inserção de pontos de Bowyer para triangulação Delaunay.

2.2. ALGORITMO DE AVANÇO DA FRENTE.

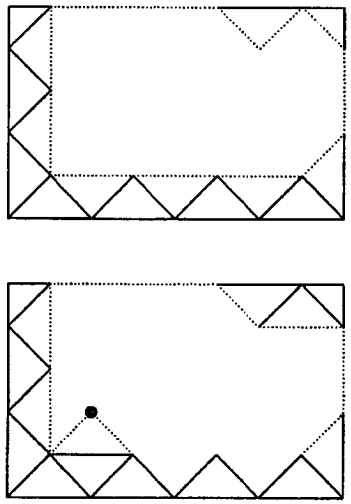


fig.2.2.1. Ilustração do conceito convencional de geração de malhas com Avanço da Frente em duas dimensões. A linha tracejada representa a corrente frente. Novos triângulos são gerados em um mesmo intervalo de tempo, através da ligação de duas extremidades de uma borda da frente até um outro ponto recentemente criado, ou um ponto presente na frente.

Técnicas de avanço da frente iniciam a discretização do modelo geométrico a partir das fronteiras, **fig.2.2.1**. Assim, atribui-se o conjunto de bordas em duas dimensões a frente inicial. Em seguida, uma borda particular desta frente é selecionada e um novo triângulo é formado, como por exemplo, com a sua base em uma das bordas. Isto é feito, ligando-se as duas extremidades desta borda até um novo ponto criado, ou até um outro ponto da frente. Em seguida, a borda selecionada deve ser removida da frente, pois está agora oculta pelo novo triângulo. Similarmente, as duas bordas restantes do novo triângulo são também adicionadas ou removidas da frente, dependendo de sua visibilidade, como mostrado na **fig.2.2.1.** A frente desta maneira constitui uma pilha, onde bordas são continuamente adicionadas ou removidas. O processo termina quando a pilha estiver vazia, isto é, quando todas as frentes estiverem ligadas umas as outras e o domínio estiver totalmente discretizado.

Todas as operações tais como verificações de intercessões com outras bordas da frente, são de caráter local, isto é, verificações de intercessões são executados em bordas adjacentes e de comprimento aproximado. Finalmente, a integridade do contorno é garantida, desde que se inicie a discretização a partir do contorno.

A complexidade de tal algoritmo é menor do que se poderia esperar, desde que nesta triangulação elementos formados nunca sejam em seguida modificados. Todos os pontos, bordas, e triângulos que existem atrás da frente atual, não necessitam ser considerados por mais tempo no processo de geração. Deste modo, a única porção ativa de dados é a frente, como a frente possui uma dimensão menor que o domínio a ser discretizado, a complexidade requerida para tal algoritmo em duas dimensões é $O(\sqrt{N})$ [5], onde N é o número final de pontos gerados na malha. Se N pontos são adicionados seqüencialmente, a complexidade é de no máximo $O(N\sqrt{N})$. Contudo, com o emprego de técnicas sofisticadas de pesquisas esta complexidade pode ser reduzida para $O(N \log N)$ [5].

As desvantagem das técnicas Avanço da Frente estão relacionadas principalmente com a sua eficiência. A fase de verificações de intersecções é particularmente uma técnica de "força bruta" para assegurar a aceitabilidade de um novo triângulo, o que é relativamente dispendioso. Adicionalmente para cada triângulo gerado, a estrutura de dados deve ser atravessada do topo a base. A complexidade para a pesquisa de pontos e bordas "próximos" é $O(\log N)$ [5]. Outro fator que contribui para o aumento do custo computacional é que técnicas de Avanço da Frente constróem a malha gerando um triângulo a cada iteração. Em uma estratégia mais eficiente, deve-se construir a malha inserindo um ponto a cada iteração. Assim, a eficiência pode ser melhorada ao determinar todos os triângulos que ligam este novo ponto a frente existente, com uma única varredura na estrutura de dados.

Encontrar triângulos aceitos e pontos "melhores", implica na determinação de uma escala de comprimento local, que defina a região de pontos e bordas "próximos". Esta escala de comprimento é geralmente obtida a partir da função esfera. Esta função fornece um parâmetro local de comprimento máximo para o raio de varredura, e é utilizado para encontrar pontos e retas intersectados, contudo, este método apresenta um problema quando consideramos o caso de duas frentes que se aproximam umas das outras, até unirem-se. Neste caso, se a função esfera varia rapidamente sobre a região entre as frentes que estão se aproximando, o tamanho relativo das bordas sobre uma frente pode ser muito maior do que aquelas na outra frente. Se uma pesquisa para verificar intersecções é iniciada desde a frente com um comprimento de escala menor, a região de bordas "próximas" pode não conter a borda apropriada, e ainda pontos da outra frente. Neste caso, falhas estão sujeitas a acontecer como mostrado na **fig. 2.2.2.**

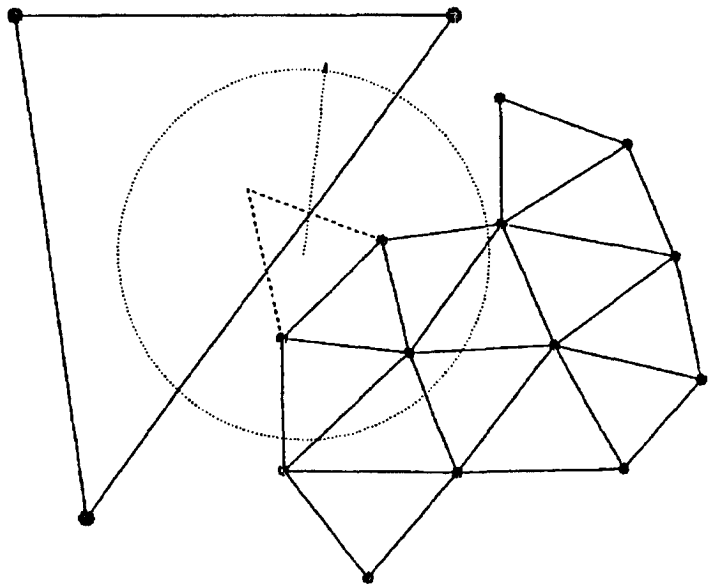


fig. 2.2.2. Ilustração de um cenário falho para o tradicional algoritmo de Avanço da Frente, onde acontece a união de duas frentes de grande diferença na escala de comprimento. O Avanço da Frente de pequenos triângulos pode falhar ao localizar os pontos das extremidades de uma grande borda nas adjacências da frente anterior ao cruzamento superior "crossover", visto que estes pontos podem estar fora da região de pesquisa definida para a pequena escala local.

CAPITULO 3

APRESENTAÇÃO DO MÉTODO

O método implementado neste trabalho, gera uma malha inicial tipo triangular irregular a partir de um contorno previamente discretizado. Este contorno discretizado, deve ser formado por segmentos de retas com comprimentos aproximados, isto para que a malha gerada a partir do mesmo não seja muito irregular. Em seguida, a malha é suavizada para tornar-se o mais regular possível. Posteriormente, a malha suavizada pode ser alterada (refinada ou desrefinada) a cada passo de iteração do algoritmo do usuário. Entretanto, neste algoritmo a malha inicial é gerada inserindo-se pontos entre cada par de arestas, selecionadas segundo um critério de menor ângulo interno entre cada par de arestas, e também segundo um parâmetro de comprimento máximo permitido para as arestas dos triângulos da malha. O algoritmo de refinamento permite alterações após sua geração sem a utilização de um processo de interpolação. A facilidade para se alterar a malha inicial é proporcionado pela aplicação de um algoritmo rápido de refinamento apenas nas células selecionadas, e também pelo armazenamento destas células num arquivo de dados para que se possa recuperá-las posteriormente.

Desta forma, primeiro o usuário fornece um contorno do domínio que se quer discretizar, e em seguida a malha inicial é gerada e suavizada. Depois, o usuário de posse desta malha inicial, determina para cada célula qual o grau de refinamento que deve ser aplicado. Em seguida, é aplicado à malha inicial o algoritmo de refinamento, segundo as especificações anteriormente determinadas pelo usuário. Por ultimo, a malha refinada é retornada ao usuário, e as alterações na malha inicial são armazenadas em um arquivo de dados.

O algoritmo proposto, consiste de 4 passos:

1. Discretizar o contorno fornecido pelo usuário segundo um parâmetro de comprimento máximo para cada sub-segmento de reta,
2. Gerar a malha inicial segundo um parâmetro de comprimento máximo para os lados dos triângulos.
3. Aplicar o processo de suavização a malha inicial para obter-se uma proporção nas aresta em torno de cada vértice interno a malha.
4. Fazer o refinamento ou a recuperação de células refinadas (desrefinamento) cada vez que o usuário desejar.

Detalhes sobre cada um destes passos serão dados nas seções seguintes.

3.1. DISCRETIZAÇÃO DO CONTORNO.

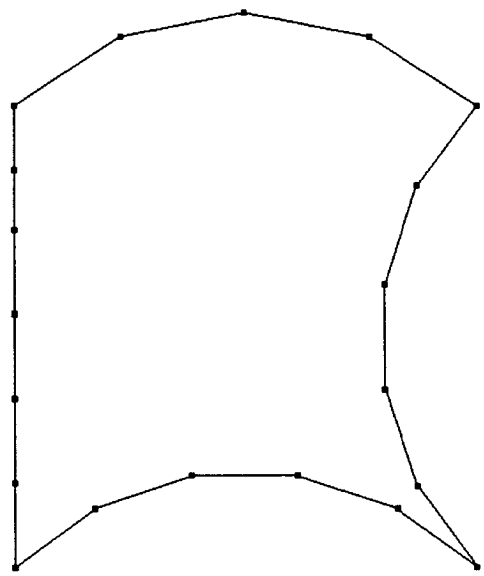


fig. 3.1 Contorno discretizado

Neste trabalho, o contorno inicial é definido como um conjunto de segmentos de retas e/ou arcos que são armazenados em uma lista encadeada. Em seguida, este contorno inicial é discretizado por seguimentos

de retas para gerar um contorno discreto, assim como o visto na **fig. 3.1**. A seqüência com que estes segmentos devem ser fornecidos pelos usuários, deve refletir a relação de vizinhança entre os mesmos. Assim, um segmento deve estar disposto na lista entre o segmento com uma extremidade coincidente com a origem do segmento em questão, e o segmento com a origem coincidente com a extremidade deste mesmo segmento.

Este segmentos iniciais são posteriormente discretizados, ou seja, são subdivididos em um número n de novos segmentos de retas. Esta divisão é feita segundo um parâmetro ρ de comprimento máximo permitido para os subsegmentos gerados. Ao discretizarmos estes segmentos, geramos uma seqüência de segmentos de retas com as mesmas propriedades da seqüência de segmentos original. Os segmentos discretizados são trocados por estas seqüências de segmentos de retas resultantes da discretização e o contorno discretizado é, desta forma, gerado. Este contorno discretizado, contudo, é composto apenas por segmentos de retas, assim como um polígono. O contorno inicial deve ser fornecido pelo usuário através de um arquivo de dados, onde devem estar presentes os seguintes dados:

- O número de segmentos de retas e ou arcos que compõem o contorno inicial,
- Os dados referentes a cada segmento que compõe o contorno inicial.

Para cada segmento do contorno deve ser fornecido o valor *0* ou *1* para a variável δ, que especifica a necessidade ou não de discretização. Neste caso, o valor *1* determina que o segmento será discretizado, e o valor *0* determina que o segmento não será discretizado. Entretanto, quando $\delta = 1$, apontar a necessidade de dizcretização, o usuário deve fornecer também um valor para ρ.

Desta maneira, o arquivo inicial do contorno terá o seguinte formato:

4

1 1 2 2 2 6 0 1 0.9

2 2 4 4 2 6 -90 0 1 0.9

3 2 8 4 6 6 90 0 1 0.9

4 2 4 4 6 2 -90 0 1 0.9

Onde na primeira linha tem-se o número de segmentos iniciais que compõem o contorno, e nas linhas seguintes tem-se os dados referentes a cada segmento. Por exemplo, na segunda linha tem-se o segmento de reta

1 1 2 2 2 6 0 1 0.9,

e da esquerda para a direita tem-se os seguintes dados:

- O número do segmento = 1
- O tipo do segmento = 1 (no caso um segmento de reta)
- As componentes *(x = 2, y = 2)* do ponto da origem do segmento de reta
- As componentes *(x = 2, y = 6)* do ponto da extremidade do segmento de reta
- O número do vizinho = 0 (neste caso não existe um triângulo vizinho)
- A variável de discretização = 1 (que determina a necessidade de discretização)
- O parâmetro de discretização ρ = 0.9 (que deve ser menor que o comprimento do seguimento de reta que se deseja discretizar)

Na terceira linha tem-se o arco

2 4 4 2 6 -90 0 1 0.9,

e da esquerda para a direita temos os seguintes dados:

- O número do segmento = 2

- O tipo do segmento = 2 (no caso um arco)
- As componentes *(x = 4, y = 4)* do ponto da origem do raio que percorre o arco
- As componentes *(x = 2, y = 6)* do ponto da extremidade do mesmo raio
- O ângulo ϕ = -90° que deve ser varrido pelo raio (o sinal de menos determina um sentido anti-horário)
- O número do vizinho = 0 (neste caso não existe um triângulo vizinho)
- A variável de discretização = 1 (que determina a necessidade de discretização)
- O parâmetro de discretização = 0.9 (que deve ser menor que o comprimento do seguimento de reta secante ao arco que se deseja discretizar)

O processo de discretização de um segmento de reta se resume nos seguintes passos:

1. Encontra-se o número *n* de segmentos de retas de comprimento menor ou igual a ρ, que são necessários para substituir o segmento inicial discretizado. O número *n* é obtido, dividindo-se o comprimento do segmento por ρ.
2. Encontra-se um vetor ***n*** normal ao segmento de reta que deverá ser discretizado.
3. Multiplica-se este vetor ***n*** por ρ, obtendo-se assim um vetor ρ, com módulo igual a ρ.
4. Iguala-se o ponto *p3* ao ponto da extremidade do segmento que está sendo discretizado.
5. Iguala-se o ponto *p1* ao ponto p da origem do segmento que será discretizado.

6. Gera-se um novo segmento de reta.

7. Atribui-se *p1* a origem do novo sub-segmento.

8. Adiciona-se ao ponto *p1* as projeções de ρ, gerando assim um novo ponto *p2*.

9. Atribui-se *p2* a extremidade do novo segmento.

10. Iguala-se *p2* a *p1*.

11. Enquanto $p2 \neq p3$, retorna-se ao passo 5.

Um arco é discretizado de uma maneira similar, entretanto, para um arco deve-se fazer o segmento de reta referente ao raio varrer um ângulo φ, **fig.3.1.2**, cuja secante tenha um comprimento menor ou igual a ρ, isto a cada geração de um novo segmento de reta resultante da discretização. O processo de discretização de um segmento de reta pode ser resumido aos seguintes passos:

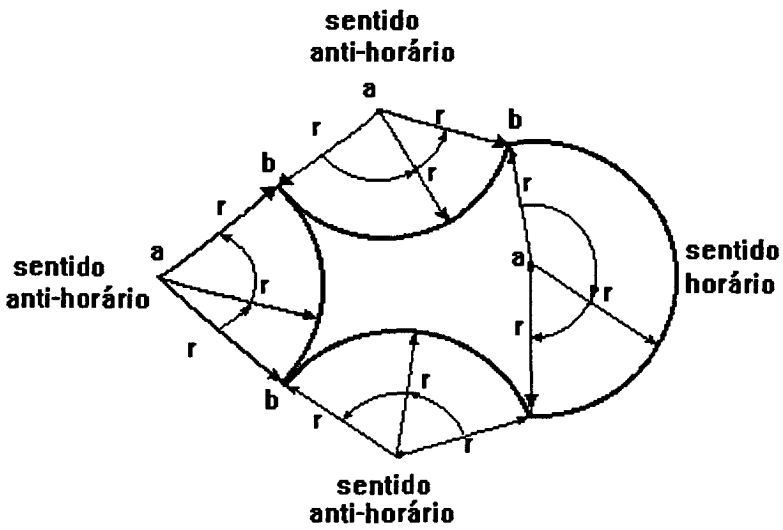


fig.3.1.2 Sentido de varredura do raio **r** para gerar o contorno discretizado

1. Obtém-se um angulo φ cuja a secante seja menor ou igual a ρ.

2. Iguala-se o ponto *p1* ao ponto *p* da extremidade do segmento de reta que representa o raio.
3. Gira-se o segmento reta referente ao raio de um ângulo ϕ, obtendo assim um ponto *p3* em sua extremidade. Este ponto é a extremidade do arco em questão.
4. Gera-se um novo segmento de reta.
5. Atribui-se *p1* a origem do novo segmento de reta.
6. Girar o raio de um ângulo φ.
7. Atribui-se ao ponto *p2* a extremidade do raio girado de φ.
8. Atribui-se *p2* a extremidade do novo segmento de reta.
9. Iguala-se *p1* a *p2*.
10. E iguala-se o raio ao novo segmento de reta.
11. Enquanto $p2 \neq p3$, retorne ao passo 4.

3.2. GERAÇÃO DA MALHA.

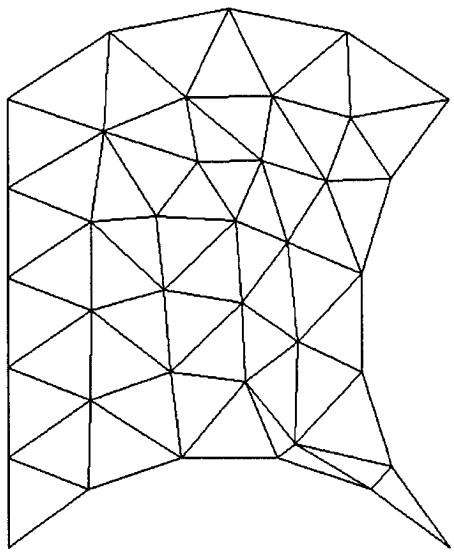


fig.3.2 Malha Inicial

A discretização é o item mais importante deste trabalho. Procurou-se obter um método de discretização eficiente e robusto para as finalidades propostas. Portanto, como já fora mencionado nos capítulos anteriores, foram pesquisados diferentes métodos de discretização e chegou-se a conclusão que, um método tipo Avanço da Frente que facilitasse o refinamento e a recuperação das células refinadas, deveria ser adequado para os tipos de problemas que este trabalho pretende prestar-se a discretizar.

Desta forma, uma variação do método de Avanço da Frente foi desenvolvida. A diferença deste método, é que sempre procura-se preencher primeiro o par de aresta com o menor ângulo interno a frente. Isto implica em sempre avançar a frente de maneira a preencher primeiro os cantos, ou seja, as regiões mais côncavas (vistas pelo lado de dentro do contorno), e apenas no final da geração deve ocorrer o preenchimento da região central, ou seja, as regiões menos côncavas.

É importante salientar que este método propõe gerar um malha inicial espaçada e o mais regular possível, assim sendo, a concentração de pontos em regiões específicas só deve ser feita após a malha inicial ter sido gerada. De outra forma, o algoritmo pode não garantir a integridade do contorno durante o processo de geração da malha inicial.

A geração da malha inicial é feita da seguinte maneira: Primeiro, atribui-se à frente o contorno previamente discretizado. Em seguida, procura-se na frente um par de segmentos de retas (arestas) vizinhos entre si, que possuam um ângulo interno a frente atual $\theta <= \beta$, onde $\beta = 90^o$ é o *ângulo crítico*. Se este não existir, então deve-se procurar o menor ângulo θ entre cada par de arestas vizinhos, assim sucessivamente, até que sobrem apenas três arestas na frente. Neste último caso, insere-se o ultimo triângulo na malha e apaga-se a frente.

Sendo assim, existem apenas três casos distintos:

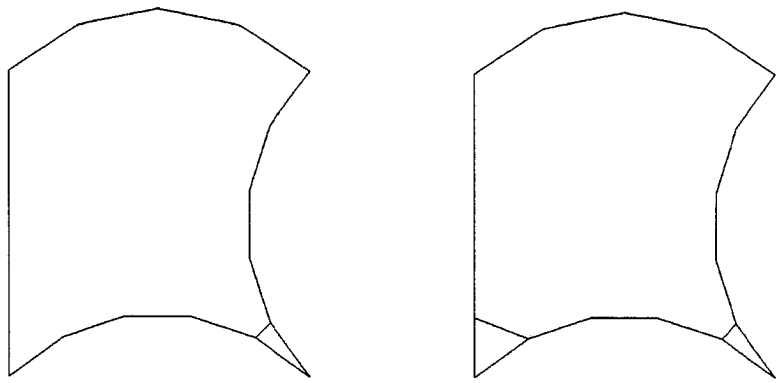


fig.3.2.1 Inserção de triângulos nas bordas com ângulo menor ou igual ao ângulo crítico.

1. Quando o ângulo interno é $\theta <= \beta$ (mostrado acima na **fig.3.2.1**). Neste caso, é necessário inserir apenas um triângulo. Assim, insere-se um triângulo com os lados coincidentes com o par de segmentos de retas que foi selecionado pelo critério do ângulo $\theta <= \beta$. Em seguida, apaga-se da frente este par de segmentos de retas, e insere-se um segmento de reta coincidente com o último lado do triângulo.

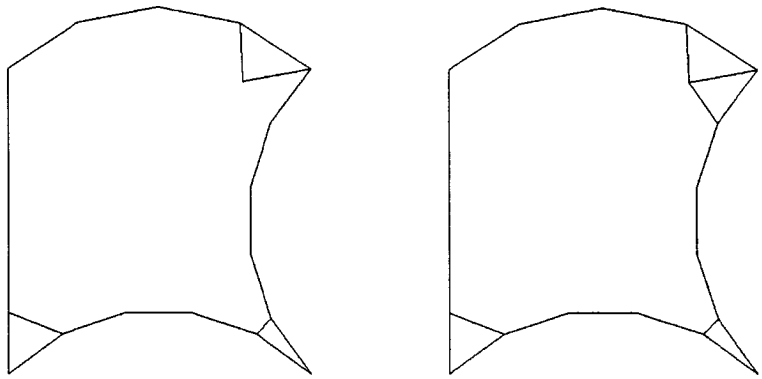


fig.3.2.2 Inserção de triângulos nas bordas com ângulos maiores que o ângulo crítico

2. Quando o ângulo interno é $\theta > \beta$ (mostrado acima na **fig.3.2.2**), torna-se necessário inserir mais de um triângulo. Neste caso, encontra-se o menor ângulo $\phi <= \beta$, cujo θ seja um de seus múltiplos. Posteriormente, varia-se

$\psi = \phi \ldots (\theta - \phi)$, e calcula-se o novo ponto, ao girar de um angulo ψ o segmento de reta da frente onde será inserido o triângulo. Depois, insere-se um triângulo com seu primeiro lado coincidente com o segmento de reta anterior, e com o segundo lado coincidente com o segmento de reta que acabou de ser gerado. Então, apaga-se o segmento de reta anterior e insere-se um novo segmento de reta, coincidente com o último lado livre do novo triângulo gerado. Procede-se desta maneira sucessivamente até chegar-se a $\phi = \theta$, em seguida volta-se a proceder da mesma forma que no primeiro caso.

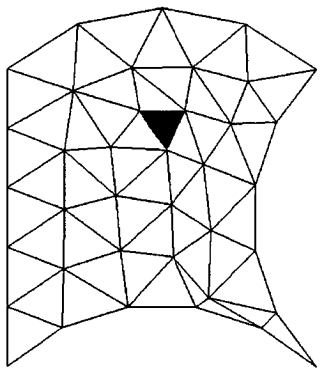


fig.3.2.3 Inserção do último triângulo nas últimas três bordas

3. Quando restarem apenas três segmentos de retas, como mostrado na **fig.3.2.3**, o último triângulo deve ser inserido. Neste caso, também procede-se de maneira similar ao primeiro caso, exceto pelo fato de se ter que inserir o triângulo entre três segmentos de retas ao invés de dois, como no primeiro caso, e também porque apagam-se os três últimos segmentos de retas da frente ao invés de apenas dois, como no primeiro caso.

Como é observado na **fig.3.2,** esta triangulação não é ideal e existem regiões onde ocorre uma desproporção entre alguns triângulos vizinhos entre si. Esta desproporção pode ser bastante reduzida se deslocarmos alguns pontos na malha, ou seja, se aplicarmos um processo de suavização, assim como o mostrado na seção seguinte.

3.3. SUAVIZAÇÃO DA MALHA.

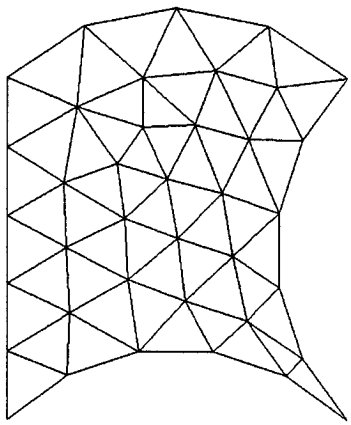


fig.3.3 Malha Suavizada

A suavização da malha é necessária para transformar um malha inicial **fig.3.2,** onde existem triângulos com arestas muito pequenas ao lado de triângulos com arestas muito grandes, em uma malha com células com dimensões mais uniformes, **fig.3.3**.

O método de suavização utilizado, calcula o centro de um polígono a partir de cada ponto central C_T de todos os triângulos que contenham um ponto (vértice) em comum C_p, **fig.3.3.2**. Portanto, é preciso gerar uma lista contendo cada ponto C_p interno a malha, e também cada sub-lista de todos os triângulos que tem como vértice C_p . Desta forma, varre-se a sub-lista com todos os triângulos que contém um determinado ponto C_p internos a malha, e aplica-se a todos estes triângulos a **eq.3.3.1** para encontrar o centro exato $C_p^{\,l}$ do polígono em questão:

$$C_p^{\,l} = \frac{\sum_{i=1}^{N} C_{t_i} A_i}{\sum_{i=1}^{N} A_i} \qquad \textbf{(eq.3.3.1)}$$

onde:

•A é a área de cada triângulo

•C_t é o centro de cada triângulo

•C_p^I é o novo centro do polígono.

A equação anterior, retorna o baricentro C_p^I do polígono que é formado por cada triângulo que contém o ponto C_p interno ao polígono. Este ponto C_p^I deverá substituir o ponto inicial C_p, provocando uma minimização das diferenças entre os tamanho das arestas internas a cada polígono. Esta minimização implica na suavização da malha, ou seja, minimiza também as diferenças entre as arestas dos triângulos da malha.

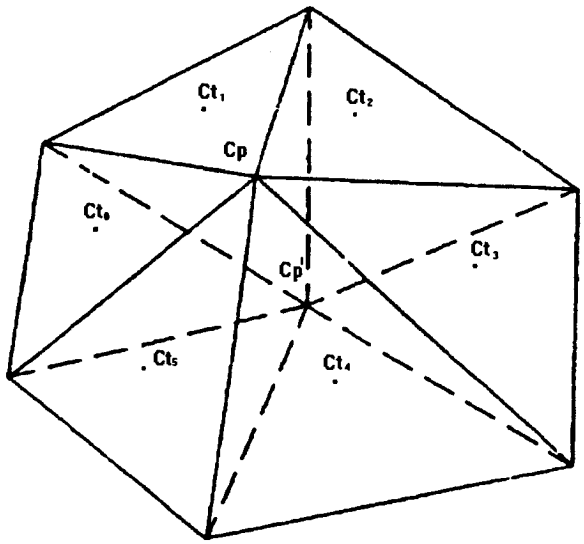


fig.3.3.1 Suavização em torno de um nó interno a malha

Auada e Meneghini [18] utilizaram um fator de distorção para as arestas dos triângulos das malhas para analisar a qualidade da malha antes e depois deste tipo de refinamento, e constatam a melhoria proporcionada por este processo de suavização. Riemslagh [10] utiliza o mesmo método de suavização utilizado neste trabalho, e afirma que em poucos passos de iteração deste algoritmo, a qualidade da malha é bastante melhorada. O autor constatou também que depois da suavização a propriedade Delaunay é restaurada.

No decorrer do desenvolvimento deste trabalho foi proposto a utilização de uma equação de sobre-relaxação **(eq.3.3.2)**, variando seu parâmetro de sobre-relaxação α de forma a se conseguir um valor ótimo C^{II}.

$$C_p^{II} = \alpha\, C_p^{I} + (1-\alpha)\, C_p \qquad \textbf{(eq.3.3.2)}$$

Entretanto, uma vez observado que $\alpha = 1$ sempre aparecia como sendo o valor ótimo para todos os casos testados. Valores $\alpha \neq 1$ acabavam por produzir polígonos tão desproporcionais ou piores que os originais. Logo, para os casos em que a malha é gerada a partir de um contorno cujas bordas possuam um comprimento aproximadamente constante, basta encontrar o centro exato de cada polígono.

3.4. REFINAMENTO DA MALHA.

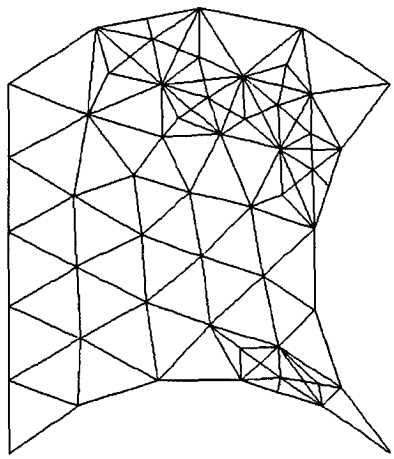


fig.3.4 Malha Refinada

O refinamento proposto neste trabalho permite ao usuário refinar um ou mais triângulos a qualquer momento no processo de iteração do seu algoritmo, sem o custoso processo de recriar toda a malha. Neste método, guardam-se os triângulos alterados a partir da malha inicial, o que implica na não necessidade de interpolações para recuperação da malha inicial.

O método de refinamento proposto consiste de três etapas onde são feitos dois tipos diferentes de refinamento de triângulos. O primeiro tipo seria um pré-refinamento que divide os triângulos marcados em três novos triângulos, todos com um vértice em comum no baricentro do antigo, agora pré-refinado, como mostrado na **fig.3.4.1**.

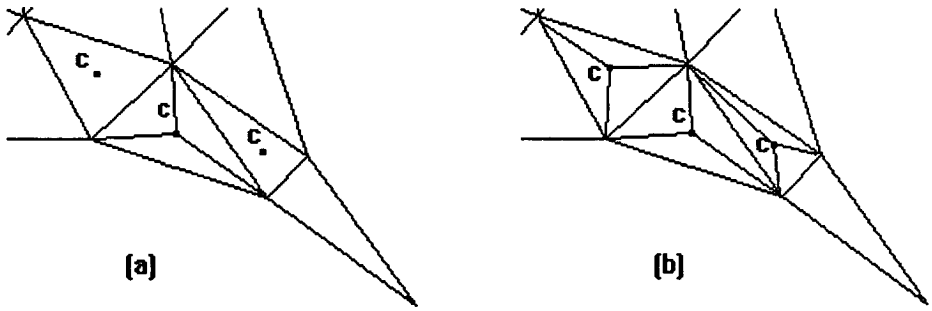


fig.3.4.1 Ampliação de um triângulo pré-refinado

O segundo tipo de refinamento consiste em determinar o centro **c** de cada aresta do triângulo marcado para o refinamento, e em seguida, colocar uma aresta que vai de cada um destes centros **c** até o centro **C** do antigo triângulo marcado, e também até cada uma dos centros **C** dos triângulos vizinhos ao triângulo marcado. Gerando-se, assim, quatro novos triângulos no lugar dos dois triângulos anteriores, como mostrado na **fig.3.4.2**. Assim, utilizaremos estas duas fases para refinar os triângulos na malha.

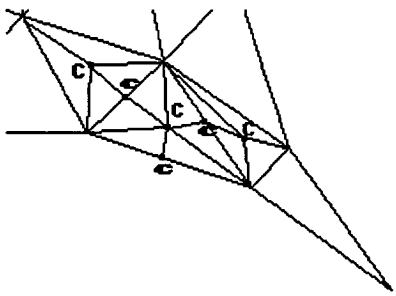


fig.3.4.2 Ampliação de triângulos pré-refinados

Estão descritos a seguir as etapas para o refinamento:

1. Aplica-se um pré-refinamento nos triângulos marcadas pelo usuário, **fig.3.4.1.a**, ao mesmo tempo que também marcam-se os triângulos vizinhos para o refinamento.
2. Aplica-se um pré-refinamento também nos triângulos vizinhos marcados no passo anterior, **fig.3.4.1.b**.
3. Refinam-se cada par (triângulo pré-refinado, vizinho pré-refinado) pelo segundo tipo de refinamento, **fig.3.4.2**.
4. Verifica-se, para cada triângulo gerado, a necessidade de se continuar o refinamento e, caso isto ocorra, marca-se cada triângulo que necessite de refinamento e retorna-se ao primeiro passo.

Naturalmente não existe limite para o grau de refinamento, entretanto, um alto grau de refinamento implica numa grande desproporção entre o tamanho das células. Assim, como já foi dito no capitulo um, se uma célula é muito menor do que sua vizinha imediata, a interface entre elas estará muito mais próxima do centro da menor célula do que da maior. Isto acarreta que, na maioria dos casos, o valor estimado estaria muito mais próximo do valor no centro da menor célula do que do valor real na interface. O que, dependendo do tipo de método numérico que se estiver usando, pode provocar erros de arredondamento.

3.4. IMPLEMENTAÇÃO.

Foram utilizadas bibliotecas desenvolvidas por terceiros, tais como: STL(Standard Template Library) [19] que faz parte do padrão ANSI/ISO para C++, desenvolvida por Alexander Stepanov. Esta é uma biblioteca que contém algoritmos para manipulação de estruturas de dados, tais como listas encadeadas, árvores binárias e outras. Desta biblioteca se utilizou exclusivamente a list.h, que contém todas as ferramentas necessárias para a manipulação de uma lista, e ainda, um campo para dados do tipo template, ou seja, o tipo de dado da lista é definido a *posteriori* pelo usuário. Outra biblioteca que foi de fundamental importância foi a geometria.h, desenvolvida por Jerônimo dos Santos Travelho. Esta biblioteca contém classes que permitem manipular informações geométricas tais como: pontos, vetores, retas, e outros, todos em duas dimensões.

Foram desenvolvidas, a partir das bibliotecas acima citadas, outras três bibliotecas. Uma primeira foi a biblioteca celula.h, desenvolvida especificamente a partir de geometria.h. A biblioteca celula.h contém uma classe virtual chamada célula, que é derivada em três outras, segmento de reta, arco e triângulo. Outra segunda biblioteca desenvolvida foi a classe boundary.h, que é basicamente uma lista de elementos do tipo ponteiro para célula. Neste caso as células são segmentos de retas e arcos, e definem o contorno usado para gerar a malha inicial. Uma terceira e ultima biblioteca desenvolvida foi a gridgene.h, que não poderia deixar de ser a biblioteca principal neste caso, pois contém todos as ferramentas para gerar, suavizar e refinar a malha. Esta classe contém também uma lista de ponteiros para célula, só que neste caso as células são sempre triângulos.

CAPITULO 4

RESULTADOS E CONCLUSÕES

Considere a malha gerada inicialmente mostrada na **fig.4.1,**

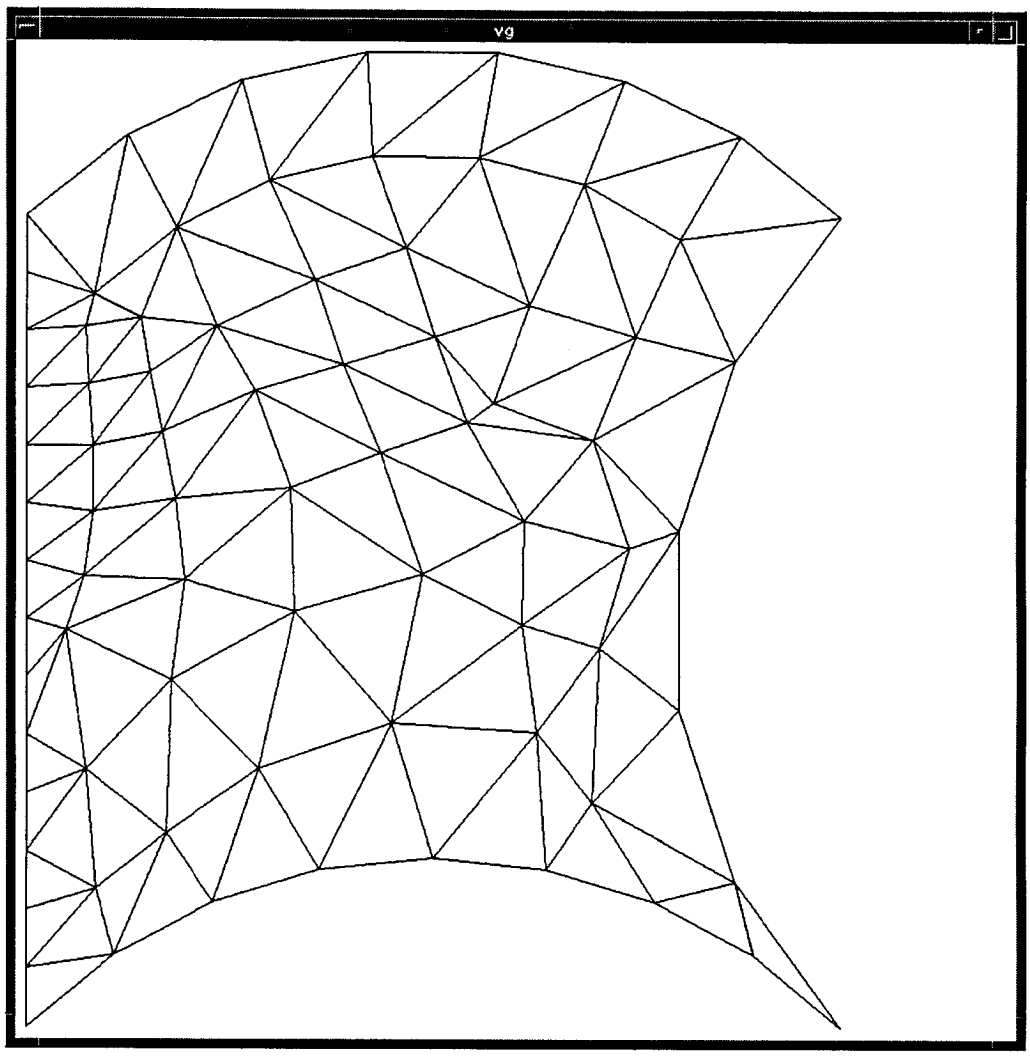


fig.4.1. Malha inicial

Após aplicação do algoritmo de suavização, obtém-se a malha suavizada, mostrada na **fig.4.2.**

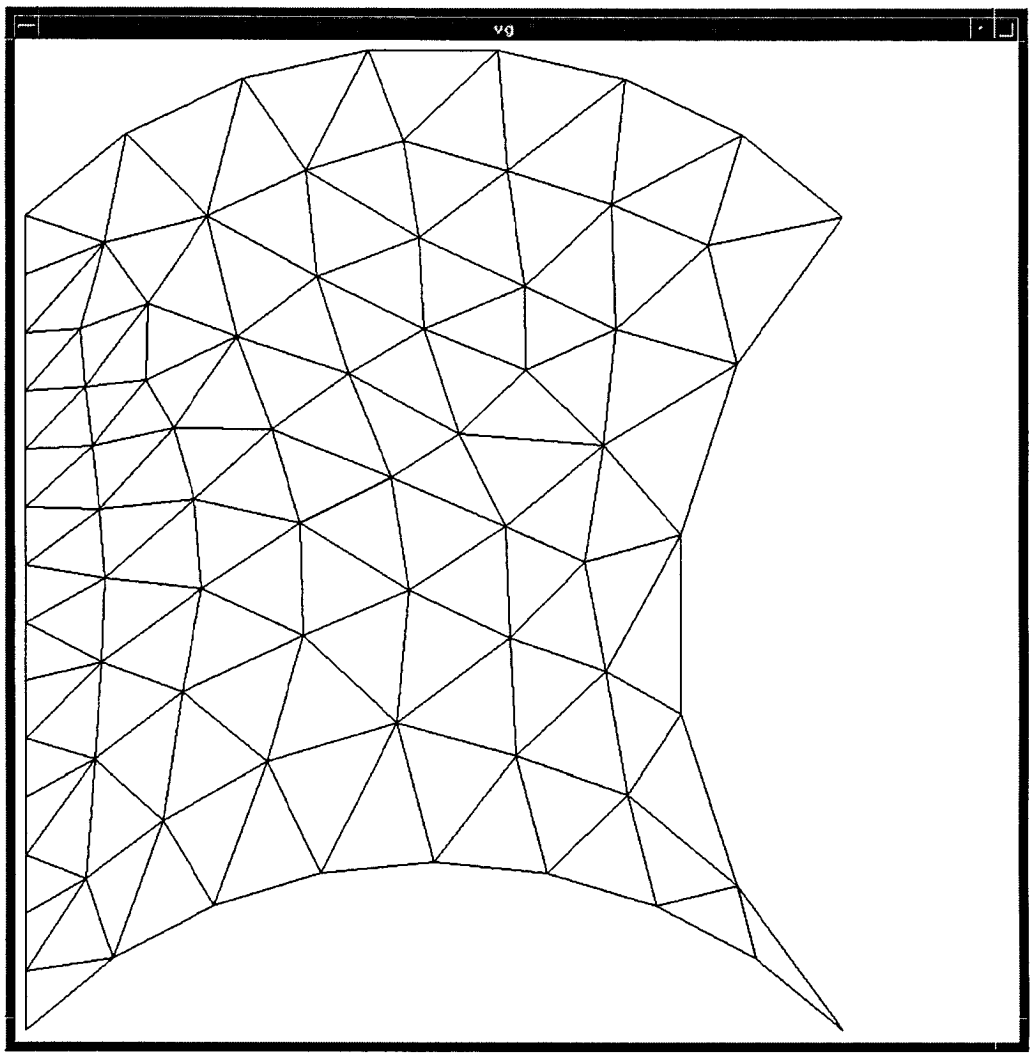


fig.4.2. Malha Suavizada

Na **fig.4.2.1,** temos a ampliação de um polígono antes e depois da aplicação do processo de suavização.

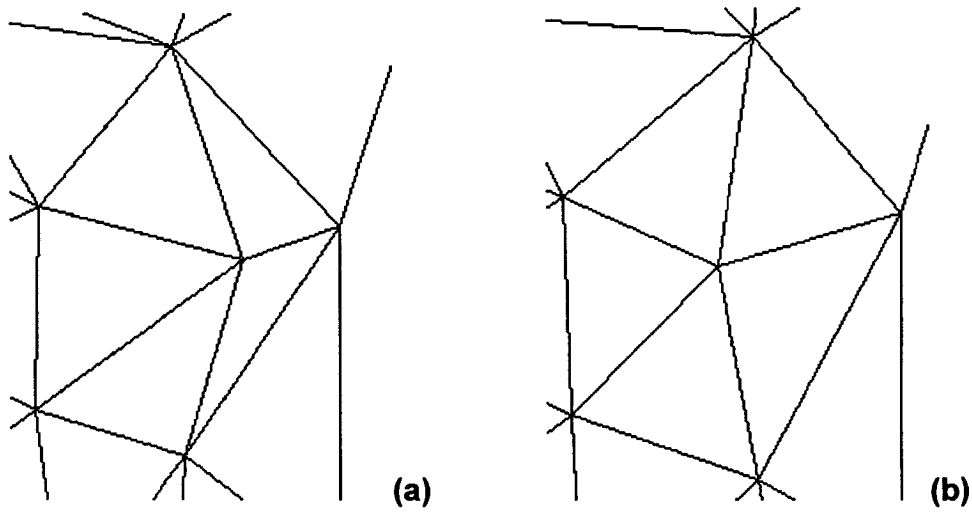


fig.4.2.1 Ampliação de um polígono antes e depois da suavização

É possível perceber numa região específica da malha inicial **fig.4.1**, ampliada na **fig.4.2.1.a**, que existe uma grande desproporção entre as arestas internas ao polígono mostrado. Também é possível observar nesta mesma região da malha suavizada, **fig.4.2**, ampliada na **fig.4.2.1.b**, que a fase de suavização corrigiu muito bem esta desproporção. Desta forma, transformou-se os triângulos deformados em triângulos mais equiláteros, gerando-se assim uma malha bem mais regular.

A **fig.4.3.1**, mostra a malha obtida após a aplicação do pré-refinamento em alguns triângulos selecionados de forma aleatória.

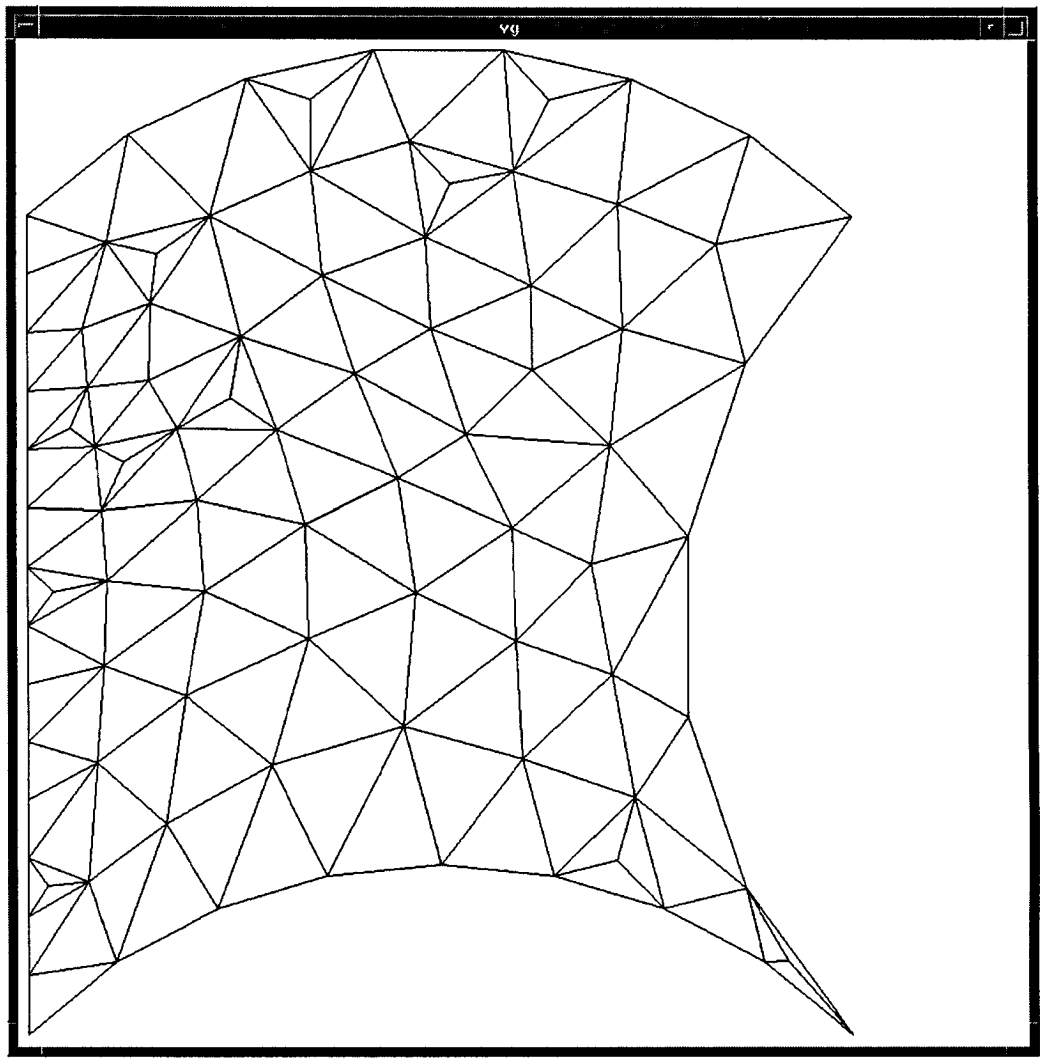
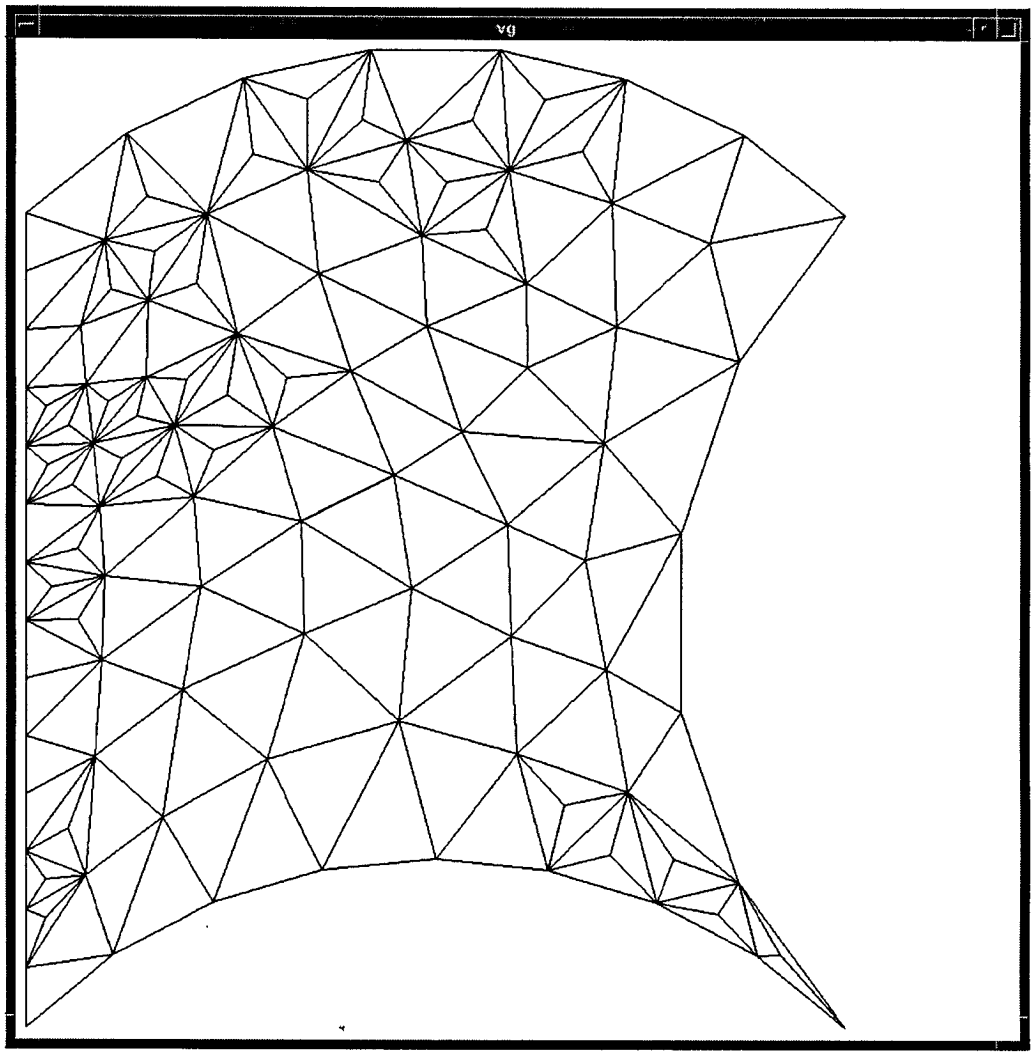


fig.4.3.1 Pré-Refinamento de Triângulos Marcados Aleatoriamente

Na **fig.4.3.2**, é mostrada a malha obtida após a aplicação do pré-refinamento nos triângulos vizinhos aos triângulos selecionados.



5.3.2. Pré-Refinamento dos Triângulos Vizinhos aos Triângulos Marcados

A **fig.4.3.3**, mostra o refinamento de cada par de triângulos (triângulo marcado e vizinho).

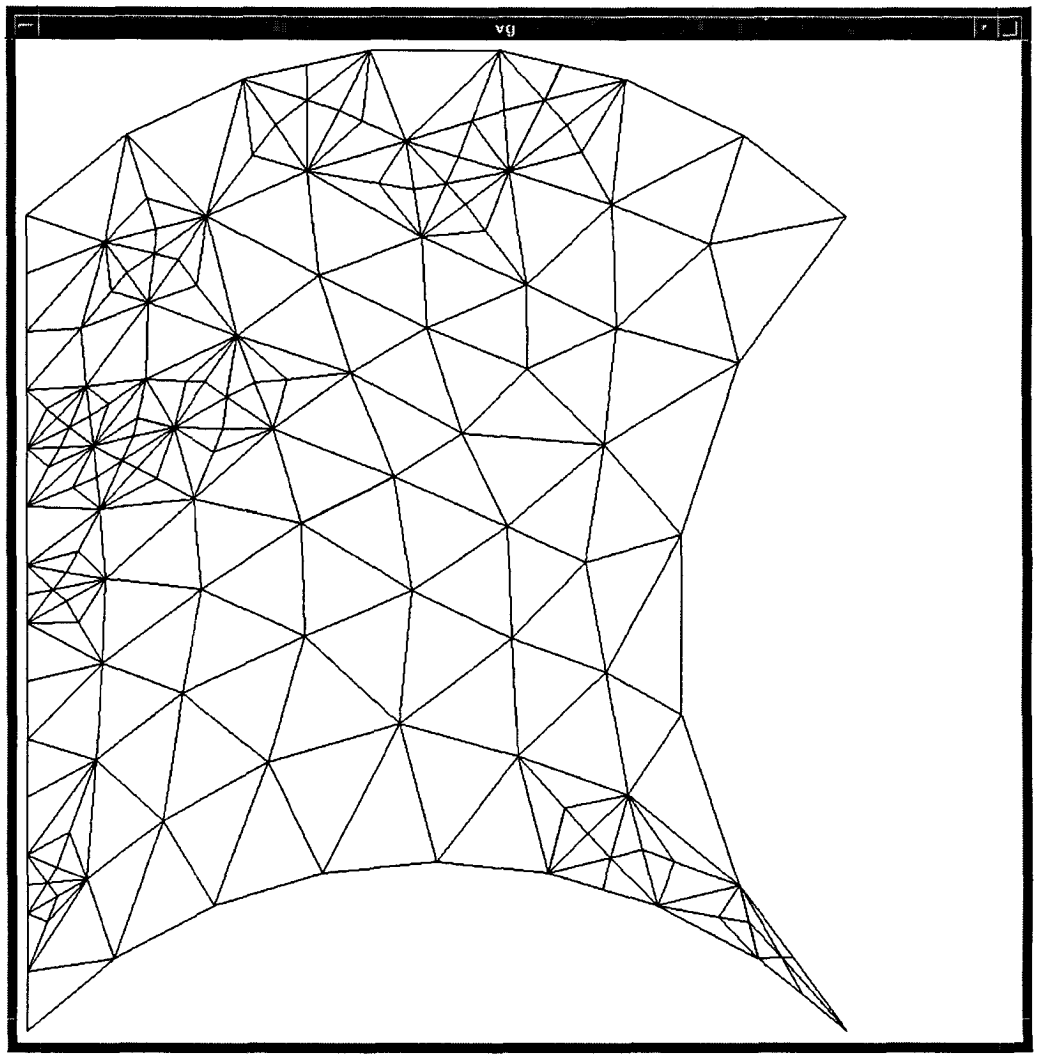


fig.4.3.3. Refinamento do Par Triângulo Marcado e Vizinho

A **fig.4.3.4**, mostra a ampliação de um triângulo marcada para o refinamento e seus vizinhos, em cada uma das fases do refinamento.

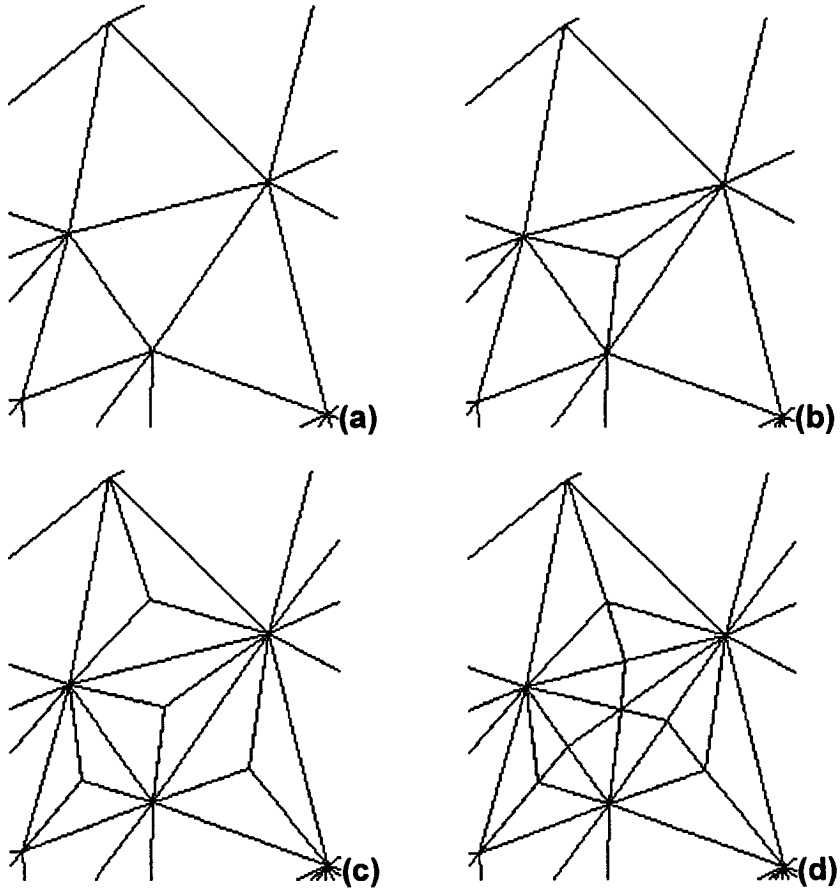


fig.4.3.4. Ampliação de um triângulo e seus vizinhos, durante as três fases do refinamento

Em **(a)** tem-se os triângulos antes do refinamento, em **(b)** tem-se os triângulo marcado já pré-refinados, em **(c)** tem-se pré-refinados os triângulos vizinhos ao triângulo marcado, e em **(d)** tem-se o refinamento dos pares de triângulos pré-refinados e vizinhos pré-refinados. Este tipo de refinamento é bastante funcional, pois, permite que qualquer célula seja subdividida até o nível desejado somente alterando os vizinhos imediatos. Desta forma, não é necessário refazer a malha toda a cada refinamento. Pode-se ainda, recuperar os triângulos que foram refinados nos passos anteriores, se armazenarmos os mesmos em um arquivo de dados.

Executando-se mais um nível de iteração no processo de refinamento, obtém-se a malha mostrada na **fig.4.3.5.**

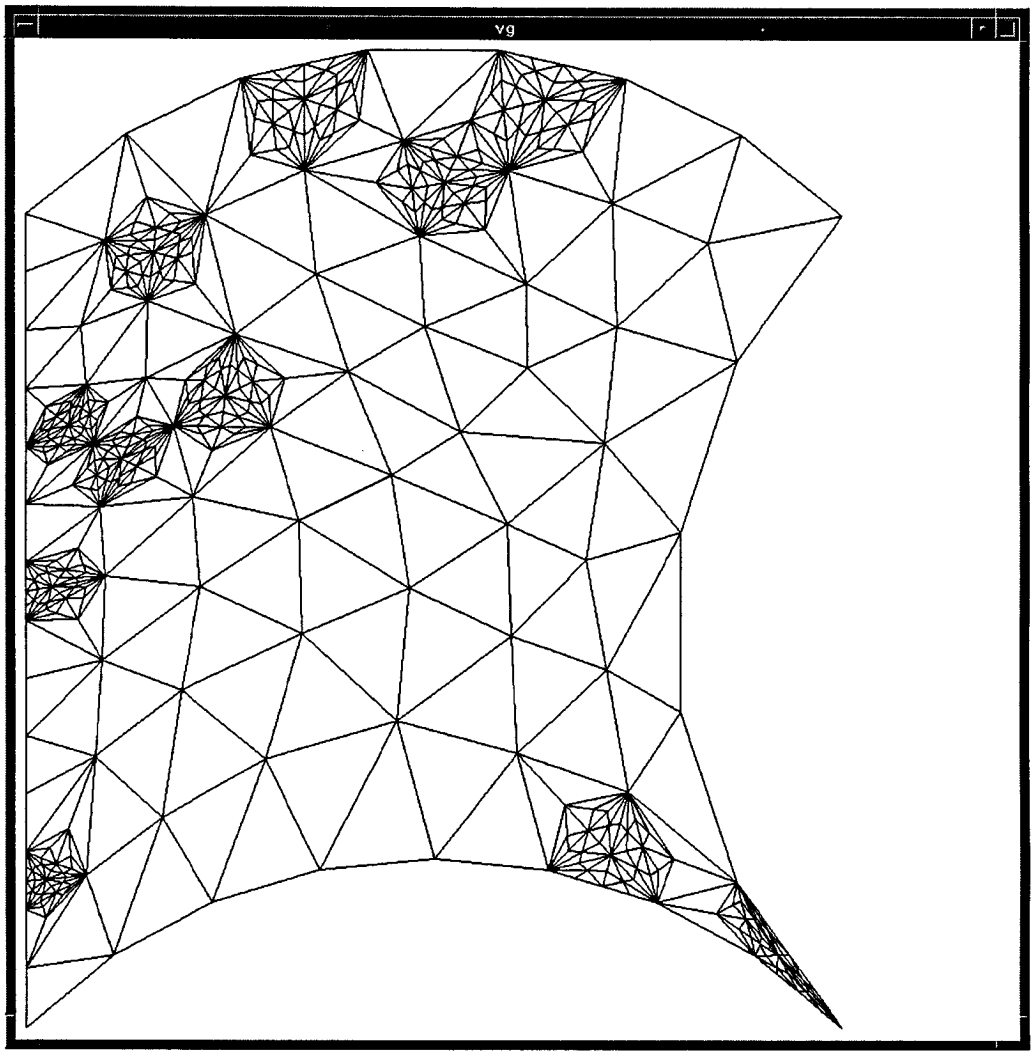


fig.4.3.5. Mais um Nível de Iteração no Processo de Refinamento

Executando-se mais dois níveis de iteração no processo de refinamento, obtém-se a malha mostrada na **fig.4.3.6.**

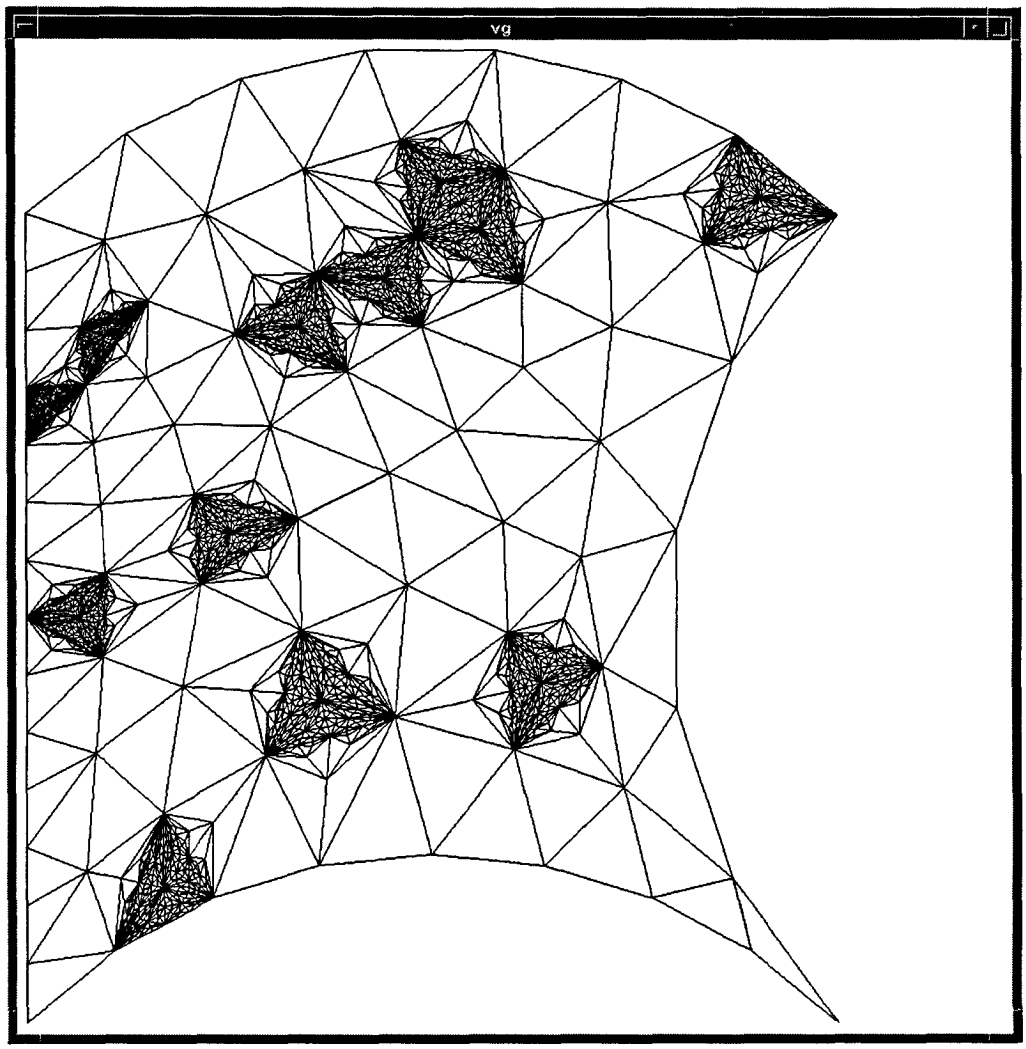


fig.4.3.6. Mais Dois Níveis de Iteração no Processo de Refinamento

A **fig.4.3.7,** mostra a ampliação de um triângulo e seus vizinhos durante algumas iterações no processo de refinamento.

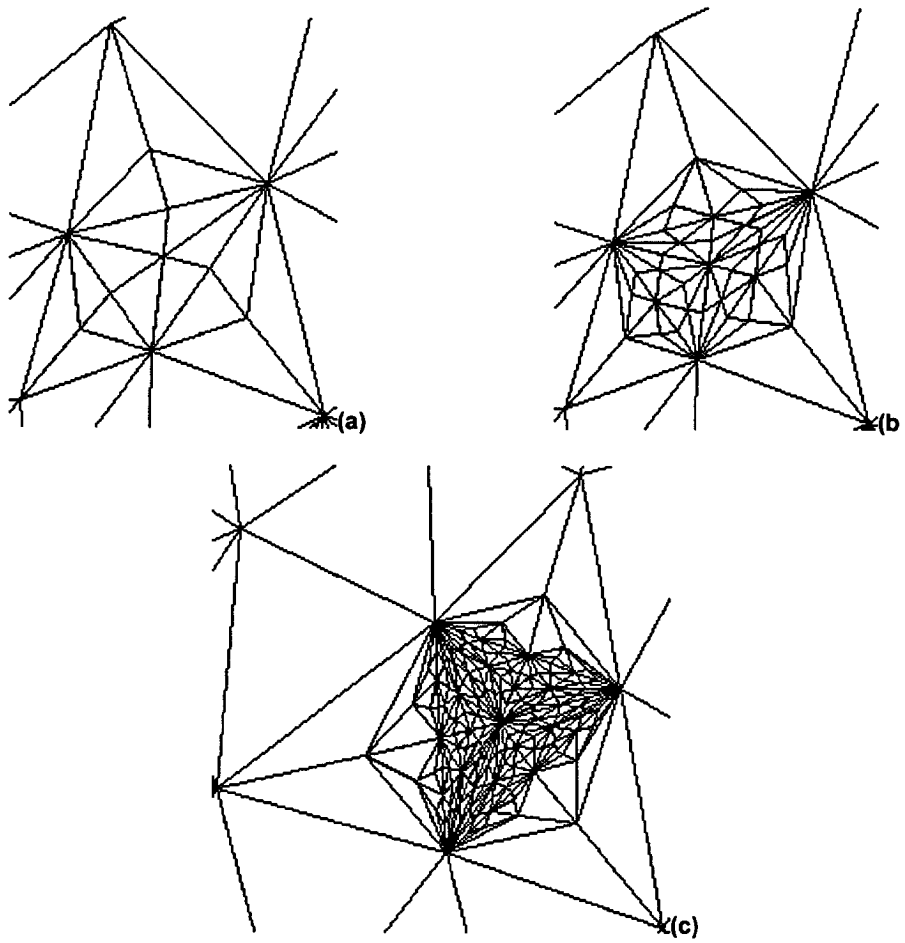


fig.4.3.7. Situação dos Triângulos Durante as Iterações do Processo de Refinamento

Como já fora mencionado anteriormente, podemos iterar indefinidamente este processo de refinamento. Entretanto, é possível perceber na **fig.4.3.8,** que os triângulos gerados pelo refinamento poderiam ser melhorados, se fosse aplicado sobre os polígonos que os contém, o mesmo processo de suavização que é utilizado após a geração da malha inicial.

CAPITULO 5

SUGESTÕES PARA TRABALHOS FUTUROS

Com relação aos tipos de contorno, podemos sugerir que sejam implementados, além de segmentos de retas e arcos de circunferências, também cônicas.

Com relação ao método de geração, podemos sugerir que seja utilizado uma função $\rho(x,y)$, ao invés de uma parâmetro constante ρ para comprimento máximo para as arestas dos triângulos gerados na malha. Esta função $\rho(x,y)$ retornaria para cada ponto com coordenada $p(x,y)$ na malha, um comprimento máximo local para as arestas dos triângulos que devem ser gerados a partir deste ponto. Isto deve permitir que os triângulos sejam condensados na região de interesse durante o processo de geração. Entretanto a determinação desta função deverá ficar por conta do usuário.

Com relação ao refinamento, podemos sugerir que seja aplicado, aos polígonos resultantes do refinamento das células, o algoritmo de suavização proposto neste trabalho. Isto pode ajudar a evitar alguma desproporção entre as arestas dos triângulos refinados e seus vizinhos não refinados. Entretanto, neste caso, seria necessário armazenar também todas as células que foram suavizadas, para poder garantir um desrefinamento como o proposto neste trabalho.

Estas sugestões não foram implementadas neste trabalho porque não faziam parte da proposta inicial.

BIBLIOGRAFIA

1. A. L. Fazenda, J. S. Travelho e M. Fabri, Modelagem Bidimensional de Solidificação Utilizando o Método de Campo de Fase, XIX CNMAC-Congresso Nacional de Matemática Aplicada e Computacional, Goiânia, setembro de 1996.
2. D.G. Kirkpatrick, Optimal search in planar subdivions, SIAM J. Comput. **12**(1), 28-35 (1983).
3. E. Edelsbrunner, L.J. Guibas e J. Stolfi, Optimal Point Location in a Monotone Subdivision, SIAM J. Comput. , **15**(2) 317-340 (1986).
4. F. P. Preparata e M. I. Shamos, Computational Geometry an Introduction, Springer-Verlag, 175 Fifth Avenue, New York Inc, New York 10010, U.S.A.
5. D. J. Mavriplis, An Advancing Front Delaunay Triangulation Algorithm Designed for Robustness, Journal of Computational Physics **117**, 90-101 (1995).
6. P. A. Burrough, 1986, Principles of Geographical Information Systems for Land Resources Assessment (Oxford:Clarendon Press).
7. Z. T. Chen, e J. A. Guevara, 1987, Systematic Selection of Very Important Points (VIP) from Digital Terrain Model for Constructing Triangular Irregular Networks. Proceedings of AUTO-CARTO 8, Baltimore, MO, U.S.A., pp.50-56.
8. M. J. MacCullagh, 1988, Terrian and Surface Modeling Systems: Theory and Practice. Photogrammetric Record, 12, 747-779.
9. C. A. Felgueiras e M. F. Goodchild, An Incremental Constrained Delaunay Triangulation, NCGIA-National Center for Geographic Information and Analysis, Technical Report 95-2, January 1995.
10. K. Riemslagh e E Dick, A Multigrid Method with Unstructured Adaptative Grids for Steady Euler Equations, Journal of Computational and Applied Mathematics 67(1996) 73-93.
11. M. Tanemura, T. Ogawa e N. Ogita, A New Algorithm for Three-Dimensional Voronoi Tesselation, J. Comput. Phys. **51** (1983) 191-207.

12.J. F. Briceño, Refinamiento de Malla Adaptable el Metodo de los Elementos Finitos, VI ENCIT-Brazillian congress of Engineering and Thermal Sciences / VI LATCYM-Latin American Heat and Mass Transfer, Florianópolis, SC Brazil (november, 1996).

13.J. L. F. Azevedo e H. Korzenowski, Utilização de Refinamento Adaptativo em Malhas Não-Estruturadas para Solução de Escoamento Hipersônicos, XIX CNMAC-Congresso Nacional de Matamética Aplicada e Computacional, Goiânia, setembro de 1996.

14.J. L. F. Azevedo e H. Korzenowski, Análise de Escoamentos Hipersônicos Utilizando Malhas Não-Estruturadas, VI ENCIT-Brazillian congress of Engineering and Thermal Sciences / VI LATCYM-Latin American Heat and Mass Transfer, Florianópolis, SC Brazil (november, 1996).

15.W. M. C. Dourado e J. L. F. Azevedo, Simulação de Escoamentos com Malhas Não-Estruturadas sobre Configurações Automotivas Básicas com um Método para Toda a Faixa de Velocidade, VI ENCIT-Brazillian congress of Engineering and Thermal Sciences / VI LATCYM-Latin American Heat and Mass Transfer, Florianópolis, SC Brazil (november, 1996).

16.S. Mathur, J. Y. Murthy e D. Choudhury, Advances in CFD Software for Heat Transfer and Combustion Calculation, VI ENCIT-Brazillian congress of Engineering and Thermal Sciences / VI LATCYM-Latin American Heat and Mass Transfer, Florianópolis, SC Brazil (november, 1996).

17.C. L. Lawson, Generation of a Triangular Grid with Application to Contour Plotting, Technical Report 299, CallTech, 1972.

18.R. B. Auada e J. M. Meneghini, Unstructured Mesh Generation: Quality Enhancement Through Smoothing, VI ENCIT-Brazillian congress of Engineering and Thermal Sciences / VI LATCYM-Latin American Heat and Mass Transfer, Florianópolis, SC Brazil (november, 1996).

19.D. R. Musser e A. Saini, C++ Programming with the Standard Template Library, Addison-Wesley Publishing Company, Reading, Massachusetts 01867.