Imprimir Fechar

Referência Completa

Tipo da Referência	Conference Proceedings
Chave Secundária	INPE-11563-PRE/6951
Chave de Citação	DemétrioTrav:2004:UnBiTr
Autor	Demétrio, Jorge Luiz Travelho, Jerônimo dos Santos
Título	Gerador de malhas para válvula de retenção em geometrias: unidimensional, bidimensional e tridimensional
Nome do Evento	Seminário de Iniciação Cientifica do INPE (SICINPE)
Localização do Evento	São José dos Campos
Ano	2004
Páginas	1-85
Título do Livro	Anais
Data	03 - 04 ago.
Editora (Publisher)	INPE
Cidade da Editora	São José dos Campos
Organização	Instituto Nacional de Pesquisas Espaciais
Repositório do Metadado	sid.inpe.br/marciana/2004/11.19.15.53.09
Palavras-Chave	COMPUTAÇÃO APLICADA, Gerador de malhas (matemática), Válvulas, Geometria, C (linguagem de programação), Mecânica dos fluidos, COMPUTER SCIENCE, Mesh generation (mathematics), Valves, Geometry, C (programming language), Fluid mechanics.
Resumo	O presente trabalho teve seu início em agosto de 2003, sendo desenvolvido com recursos PIBIC, Programa Institucional de Bolsa de Iniciação Científica, nas instalações do INPE - Instituto Nacional de Pesquisa Espacial, onde foi possível o desenvolvimento de atividades de pesquisa voltada à área computacional e de engenharia mecânica. O objetivo deste trabalho é desenvolver um gerador de malhas, de modo a contribuir para o desenvolvimento desta tecnologia através do entendimento do processo de construção de malhas estruturadas. Para isto foi necessário: a familiarização com o sistema operacional UNIX; além do conhecimento em programação da linguagem C, a qual foi adotada na elaboração código; e maior entendimento dos assuntos voltados aos conceitos de mecânica dos fluidos através da leitura dos textos disponíveis. Baseando-se nestes conhecimentos adquiridos neste período foi possível desenvolver um código cujas dimensões são preestabelecidas pelo usuário de forma coerente que podendo viabilizar a construção de uma malha estruturada, obedecendo às condições do formato do modelo de uma válvula de retenção, de modo que este permitisse para uma futura adaptação para um modelo físico adequado. Por se tratar de um programa complexo, nem todas as características do modelo da válvula são reproduzidas neste primeiro trabalho, fundamentalmente, pretende-se cumprir a proposta de trabalho referente a geração das geometrias dos casos unidimensional, bidimensional e tridimensional, obedecendo o contorno da válvula e o obstáculo, este considerado como um corpo rígido e dinâmico, gerando-se um arquivo que defini o posicionamento dos nós da malha. Como resultado adicional fez-se o desenvolvimento de um código, usando um modelo matemático para definir fatores como deslocamento, velocidade e aceleração em função do tempo, criando-se assim uma tabela de dados de saída. Usando métodos numéricos por meio de interpolação polinomial criou-se a possibilidade de obter as propriedades em cada nó desejado a analisar, definida pelo primeiro código. Após todo este processo foram gerados vários arquivos contendo os dados necessários para a construção dos gráficos de deslocamento, velocidade e aceleração, onde poderá ser feita uma analise do escoamento. O cálculo do escoamento e a visualização dos resultados poderão ser feitos em propostas para futuros trabalhos. Futuramente pretende-se aprimorar o programa, adicionando-se novas características físicas e computacionais, que o tornarão mais geral em condições de simular uma grande gama de problemas que existem e que possam surgir nesta área, a implementação do programa é feita dentro de uma proposta maior de desenvolvimento de um código computacional para analise de escoamento.
Ultima Atualização do Metadado	2008:01.08.16.31.43 sid.inpe.br/banon/2003/08.15.17.40 marciana
Site	<mtc-ml6.sid.inpe.br>

Idioma	pt
Detentor da Cópia	SID/SCD
Tipo Secundário	PRE CN
Formato	Papel
Area	COMP
Grupo	LAC-CTE-INPE-BR LAC-CTE-INPE-BR
Afiliação	Universidade de Taubaté (UNITAU) Instituto Nacional de Pesquisas Espaciais. Laboratório Associado de Computação e Matemática Aplicada (INPE.LAC)
Projeto	Computação científica
Permissão de Leitura	allow from all
e-Mail (login)	marciana
Grupo de Usuários	sergio administrator marciana
Visibilidade	shown
Data de Acesso	08 jan. 2008
Histórico	2005-06-14 18:10:15 :: sergio -> administrator 2006-11-09 18:49:27 :: administrator -> sergio 2008-01-07 12:53:30 :: sergio -> marciana
Estágio do Documento	concluido

atualizar

Fechar

Gerador de Malhas para Válvula de Retenção em Geometrias: Unidimensional, Bidimensional E Tridimensional.

Jorge Luiz Demétrio[1] (UNITAU, Bolsista PIBIC/CNPq)
Dr. Jeronimo S. Travelho[2] (LAC/CTE/INPE)

RESUMO

O presente trabalho teve seu início em agosto de 2003, sendo desenvolvido com recursos PIBIC, Programa Institucional de Bolsa de Iniciação Científica, nas instalações do INPE – Instituto Nacional de Pesquisa Espacial, onde foi possível o desenvolvimento de atividades de pesquisa voltada à área computacional e de engenharia mecânica. O objetivo deste trabalho é desenvolver um gerador de malhas, de modo a contribuir para o desenvolvimento desta tecnologia através do entendimento do processo de construção de malhas estruturadas. Para isto foi necessário: a familiarização com o sistema operacional UNIX; além do conhecimento em programação da linguagem C, a qual foi adotada na elaboração código; e maior entendimento dos assuntos voltados aos conceitos de mecânica dos fluidos através da leitura dos textos disponíveis. Baseando-se nestes conhecimentos adquiridos neste período foi possível desenvolver um código cujas dimensões são preestabelecidas pelo usuário de forma coerente que podendo viabilizar a construção de uma malha estruturada, obedecendo às condições do formato do modelo de uma válvula de retenção, de modo que este permitisse para uma futura adaptação para um modelo físico adequado. Por se tratar de um programa complexo, nem todas as características do modelo da válvula são reproduzidas neste primeiro trabalho, fundamentalmente, pretende-se cumprir a proposta de trabalho referente a geração das geometrias dos casos unidimensional, bidimensional e tridimensional, obedecendo o contorno da válvula e o obstáculo, este considerado como um corpo rígido e dinâmico, gerando-se um arquivo que defini o posicionamento dos nós da malha. Como resultado adicional fez-se o desenvolvimento de um código, usando um modelo matemático para definir fatores como deslocamento, velocidade e aceleração em função do tempo, criando-se assim uma tabela de dados de saída. Usando métodos numéricos por meio de interpolação polinomial criou-se a possibilidade de obter as propriedades em cada nó desejado a analisar, definida pelo primeiro código. Após todo este processo foram gerados vários arquivos contendo os dados necessários para a construção dos gráficos de deslocamento, velocidade e aceleração, onde poderá ser feita uma analise do escoamento. O cálculo do escoamento e a visualização dos resultados poderão ser feitos em propostas para futuros trabalhos. Futuramente pretende-se aprimorar o programa, adicionando-se novas características físicas e computacionais, que o tornarão mais geral em condições de simular uma grande gama de problemas que existem e que possam surgir nesta área, a implementação do programa é feita dentro de uma proposta maior de desenvolvimento de um código computacional para analise de escoamento.

[1] Aluno do Curso de Engenharia Mecânica , UNITAU. **E-mail: jorge@lac.inpe.br**
[2] Pesquisador da Laboratório Associado Computação Científica e Matemática Aplicada, Coordenação Geral do Centro de Tecnologia Espaciais. **E-mail: jeff@lac.inpe.br**

AUTORIZAÇÃO PARA PUBLICAÇÃO

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

Número

PIBIC-034/04

Titulo

Gerador de Malhas para Válvula de Retenção em Geometrias: Unidimensional, Bidimensional e Tridimensional

Autor

Jorge Luiz Demétrio

Tradutor

Não há

Editor

Origem	Projeto	Série	No. de Páginas	No. de Fotos	No. de Mapas
			84	0	0

Tipo

☐ RPQ ☒ PRE ☐ NTC ☐ PRP ☐ MAN ☐ PUD ☐ TAE

Divulgação

☒ Externa ☐ Interna ☐ Reservada ☐ Lista de Distribuição Anexa

Periódico / Evento

Seminário de Iniciação Científica do INPE - SICINPE 2004

Convênio

Autorização Preliminar

__/__/__
Data

Coordenador do PIBIC
Programa Institucional de Bolsas de Iniciação Científica
Titular da Unidade - INPE

Revisão Técnica

☐ Solicitada ☐ Dispensada

Recebida __/__/__ Devolvida __/__/__

Titular de Nível "A"

Assinatura do Revisor

Revisão de Linguagem

☐ Solicitada ☐ Dispensada

Recebida __/__/__ Devolvida __/__/__

Titular de Nível "A"

Assinatura do Revisor

Autorização Final

__/__/__
Data

Marcos Dias da Silva
Coordenador de Ensino, Documentação e Programas Especiais
Titular da Unidade CEP

Palavras Chave

Gerador de Malhas - Válvula de Retenção - Geometrias

(continua na folha 2/2)

INPE - 106.2

MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

INPE-11563-PRE/6951

GERADOR DE MALHAS PARA VÁLVULA DE RETENÇÃO EM GEOMETRIAS: UNIDIMENSIONAL, BIDIMENSIONAL E TRIDIMENSIONAL

Jorge Luiz Demétrio

Relatório Final de Projeto de Iniciação Científica
(PIBIC/CNPq/INPE)

INPE
São José dos Campos
2004

MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

RELATÓRIO FINAL DE PROJETO DE INICIAÇÃO CIENTÍFICA (PIBIC/CNPq/INPE):

GERADOR DE MALHAS PARA VÁLVULA DE RETENÇÃO EM GEOMETRIAS: UNIDIMENSIONAL, BIDIMENSIONAL E TRIDIMENSIONAL.

Jorge Luiz Demétrio Autor do Projeto (UNITAU, Bolsista PIBIC/CNPq)
E-mail: jorge@lac.inpe.br

Dr. Jeronimo S. Travelho (LAC/CTE/INPE, Orientador)
E-mail: jeff@lac.inpe.br

INPE
São José dos Campos

Resumo

O presente trabalho teve seu início em agosto de 2003, sendo desenvolvido com recursos PIBIC, Programa Institucional de Bolsa de Iniciação Científica, nas instalações do INPE – Instituto Nacional de Pesquisa Espacial, onde foi possível o desenvolvimento de atividades de pesquisa voltada à área computacional e de engenharia mecânica. O objetivo deste trabalho é desenvolver um gerador de malhas, de modo a contribuir para o desenvolvimento desta tecnologia através do entendimento do processo de construção de malhas estruturadas. Para isto foi necessário: a familiarização com o sistema operacional UNIX; além do conhecimento em programação da linguagem C, a qual foi adotado na elaboração código; e maior entendimento dos assuntos voltados aos conceitos de mecânica dos fluidos através da leitura dos textos disponíveis. Baseando-se nestes conhecimentos adquiridos neste período foi possível desenvolver um código cujas dimensões são preestabelecidas pelo usuário de forma coerente que podendo viabilizar a construção de uma malha estruturada, obedecendo às condições do formato do modelo de uma válvula de retenção, de modo que este permitisse para uma futura adaptação para um modelo físico adequado. Por se tratar de um programa complexo, nem todas as características do modelo da válvula são reproduzidas neste primeiro trabalho, fundamentalmente, pretende-se cumprir a proposta de trabalho referente a geração das geometrias dos casos unidimensional, bidimensional e tridimensional, obedecendo o contorno da válvula e o obstáculo, este considerado como um corpo rígido e dinâmico, gerando-se um arquivo que defini o posicionamento dos nós da malha. Como resultado adicional fez-se o desenvolvimento de um código, usando um

modelo matemático para definir fatores como deslocamento, velocidade e aceleração em função do tempo, criando-se assim uma tabela de dados de saída. Usando métodos numéricos por meio de interpolação polinomial criou-se a possibilidade de obter as propriedades em cada nó desejado a analisar, definida pelo primeiro código. Após todo este processo foram gerados vários arquivos contendo os dados necessários para a construção dos gráficos de deslocamento, velocidade e aceleração, onde poderá ser feita uma analise do escoamento. O cálculo do escoamento e a visualização dos resultados poderão ser feitos em propostas para futuros trabalhos. Futuramente pretende-se aprimorar o programa, adicionando-se novas características físicas e computacionais, que o tornarão mais geral em condições de simular uma grande gama de problemas que existem e que possam surgir nesta área, a implementação do programa é feita dentro de uma proposta maior de desenvolvimento de um código computacional para analise de escoamento.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

SIMBOLOGIA

Arábicos

r, α, z..... Vetores de base do sistema de coordenadas cilíndricas;

i, j, k.....Vetores de base do sistema de coordenadas cartesianas;

CAP. 1 INTRODUÇÃO

1.1 Motivação:

Por se tratar de um tópico de grande interesse na indústria hidrodinâmica, o estudo dos processos de formação de vórtice em perfis hidrodinâmicos tem sido alvo de razoável investigação e o interesse nesta tecnologia vem crescendo rapidamente.

Para obter um maior conhecimento no escoamento de fluido dentro da válvula de retenção, faz se necessário simular no computador, assim podendo ser possível analisar as reações na válvula, como turbulência, força aplicada na válvula, contorno do fluido bem como analise em diversos aspectos de apresentação do fluido dentro da válvula. Tudo isto leva a necessidade de que seja feita uma malha que defina uma melhor precisão próxima ao retentor, permitindo obter as propriedades desejadas.

1.2 PROPOSTA DO TRABALHO:

Objetivo deste estudo é contribuir para o enriquecimento desta tecnologia através de um entendimento do processo de formação da malha, principalmente, no desenvolvimento de um programa gerador de malhas obedecendo aos contornos externo da válvula e o obstáculo de forma dinâmica. Pois nem todas as características do modelo poderão ser executadas neste trabalho, devido ao tempo que levaria para serem reproduzidas graficamente.

Fundamentalmente, pretende-se cumprir as seguintes etapas: o entendimento do assunto através da leitura dos textos disponíveis; a construção do código que fará o posicionamento dos nós da malha, para os três casos (unidimensional, bidimensional e tridimensional), o qual pode se adequar a vários tamanhos de válvula de retenção que poderia ser considerado pelo usuário.

E para trabalho futuro a implementação do programa dentro de uma proposta maior de desenvolvimento de um simulador para a previsão de deslocamento, velocidade aceleração e dentre outras propriedade. Se houver tempo como trabalho adicional poderá ser apresentado um simulador seria uma simples linearização, tornando a aceleração como constante e obtenção da velocidade e deslocamento por métodos matemáticos.

Objetivo maior será a geração da geometria, para uso de cálculo do escoamento e a visualização dos resultados com ferramentas, que irão fazer parte de proposta para futuro trabalho á analisar o escoamento com profundidade.

Futuramente pretende-se aprimorar o programa, adicionando novas características físicas e computacionais, que o tornarão mais generalizado. Em condições de simular uma grande gama de problemas que existem e que possam surgir nesta área.

2.1 O Fenômeno Físico:

Freqüentemente o maior problema na hidrodinâmica é ocasionado devido pela turbulência, o qual se apresenta na maioria dos casos quando o volume de controle possui um obstáculo ao escoamento, fazendo ocorrer regiões com turbulência. No interior deste sistema, as partículas do fluido que se escoava sobre um regime laminar sofrem uma perturbação e se transforma em regime turbulento, ou seja, redefinindo uma nova trajetória de recirculação chamada de vórtice, como mostra a figura 1.0.

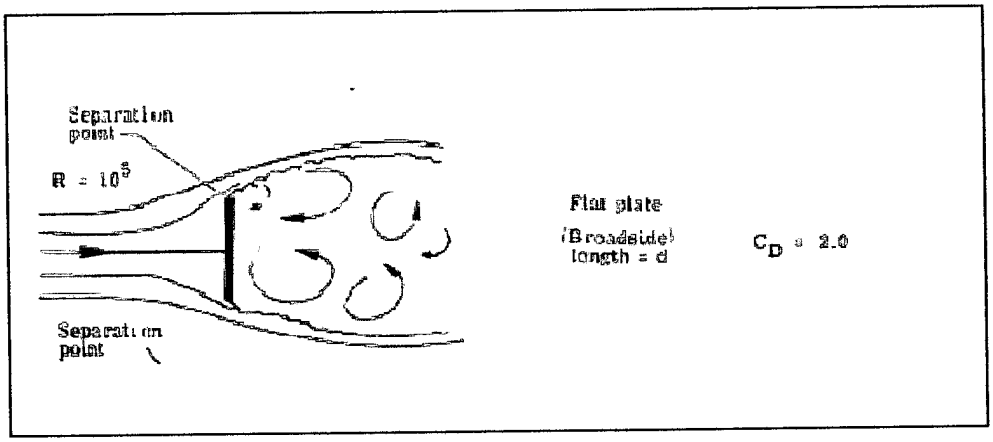

Fig. 1.0

Figura 1.0 – Extraído Introduction to the Aerodynamics of Flight.

As partículas próximas às paredes da válvula aderem à estrutura considerando que a velocidade seja nula, formando varias lâmina do fluido nas camadas superiores sobre a superfície. Assim, devido ao escoamento, ocorre um processo termodinâmico nesta lâmina envolvendo convecção, evaporação e outros fenômenos térmicos.

Além de depender da superfície acabamento interno da válvula, pois este quando aumentado a sua rugosidade superficial, este estimulo faz com que o fluido tenha maior dificuldade no escoamento próximo as paredes da válvula, podendo fazer com que a camada limite descole antes do que deveria.

Uma observação importante que deve ser feita é quanto as formação do vórtice que podem ocorrer dependendo diretamente dos parâmetros térmicos, rugosidade das paredes, viscosidade dinâmica, densidade e velocidade. Basicamente são esses os coeficientes que podem interferir nas diversas formas adotados pêlos vórtice. Além do formato do obstáculo que será decisiva para não formação do vórtice, a construção do obstáculo deve ter o formato a facilitar o escoamento, ou ocasionar a menor turbulência possível ao sistema, pois este é indesejável.

A formação do vórtice faz com que o escoamento se torne mais complexo nesta região e como será discutido logo a seguir a forma de divisão da malha, pois é caracterizado por formações de zonas de recirculação do fluido nesta região isto faz com que a malha tenha que ser refinada para uma melhor visualização na construção dos gráficos.

2.2 Descrição do modelo:

A previsão teórica para a construção gráfica da formação do vórtice na hidrodinâmica envolve basicamente quatro itens. São elas: a definição da geometria do obstáculo; a obtenção do escoamento das propriedades do fluido; o cálculo de camada limite para os efeitos viscosos e de transferência de calor; o cálculo das trajetórias das partículas do fluido que colidem com a superfície; porem como mencionado anteriormente será cumprida somente a definição da geometria da malha.

O problema é considerado para os casos unidimensional, bidimensional e tridimensional, à parte de cálculo envolvida na divisão da malha será desenvolvida de modo que todos os parâmetros necessários para a construção da geometria da malha se desenvolva de forma dinâmica.

3.1 MODELO:

O modelo matemático adotado na divisão é uma das partes mais importantes, pois irá definir o posicionamento dos nós, porém estes cálculos serão executados de forma a obter resposta para cada posicionamento do obstáculo dinâmico para cada posição inserida pelo usuário, estas posições por sua vez serão lidas de um arquivo criado pelo usuário. Tal obstáculo contido no volume de controle ainda terá como propriedade ser um corpo rígido, ou seja, não irá se deformar com a força aplicado pelo escoamento.

3.2 Definição da geometria da válvula:

O primeiro passo no desenvolvimento da solução do problema via métodos computacionais é a definição do tipo de válvula de retenção, vide figura 3.1 este é o modelo adotado, deste modelo serão considerados as paredes externas e o retentor que será dinâmico a se ter como domínio a forma, em que as equações deverão se adequar para uma formação gráfica. Por se tratar de um programa complexo, nem todas as características do modelo da válvula são reproduzidas neste primeiro trabalho, fundamentalmente pretende-se cumprir a proposta inicial de trabalho da geração da geometria dos casos unidimensional, bidimensionais e tridimensionais, obedecendo aos contornos externo impostos pela válvula e o êmbolo, considerando-o um corpo rígido e dinâmico no sistema. Como o êmbolo varia o seu posicionamento, e a cada posição tomada tem que redefinir o posicionamento de cada nó da malha. Limitando este posicionamento em

uma região de possível deslocamento, pois uma região na entrada e na saída será destinada ao encaixe da válvula de retenção na tubulação.

Tais dimensões serão criadas pelo usuário de forma coerente, pois os valores não estão limitados no código, e pode ocorre erro ao determinar valores muito grandes ou pequenos.

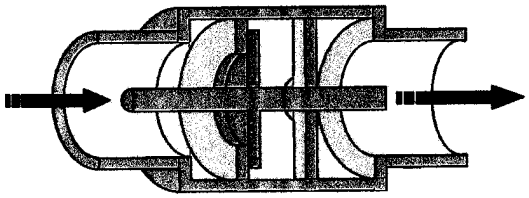

Fig. 3.1

4.1 Definições de cotas:

Para a construção da malha são necessárias as definições das cotas, que são determinadas pelo usuário, de modo a definir com coerência as medidas a serem usadas são elas: comprimento (x) e a posição do centro de gravidade do obstáculo no vetor x (x1).

Este, porém deverá ser criado um arquivo cujo nome de "Dado.txt" para a inserção destas posições, de modo em que o primeiro número deve ser inteiro e positivo determinando o numero de posições inserida no arquivo. Vide tabela 4.1.

No desenvolvimento do código fica automática determinada a espessura do obstáculo que é um décimo do comprimento no vetor x estabelecido pelo usuário.

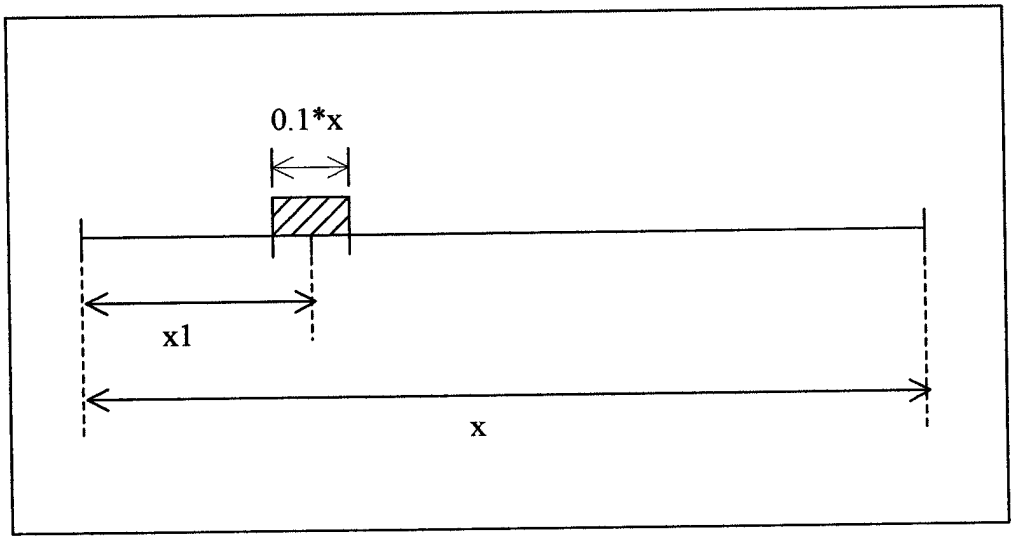

Fig. 4.1

O primeiro passo tomado pelo código é fazer a leitura do arquivo "Dado.txt", alocando um vetor com as posições desejadas, condicionando com

que estes dados não seja menor e nem maior do que os números pertencentes ao domínio dos números reais possíveis de realizar o código com sucesso.

Quando definido o tamanho do vetor, define se também o numero de divisões do vetor x, mas foi criado um fator que multiplicativo o mesmo possibilitando assim uma maior divisão do vetor x, ou seja, um refinamento do intervalo. Este fator deverá ser determinado pelo usuário de forma coerente, devendo ser um numero inteiro e positivo, pois produto será o termo de alocação do vetor x.

Os posicionamentos destes pontos serão dados de forma progressiva, com intervalo decrescente, do ponto inicial até o obstáculo e de forma progressiva, com intervalo crescente, do obstáculo até a parte final.

Após o código evoluir todos estes passos os resultados obtidos serão salvos com o nome de "Coordenada_plotar_n. Xls". Onde o "n" será definido pelo numero da interação, sendo que o arquivo possui uma extensão Excel, tais resultados serão possível plotar no Excel e visualizar estes pontos.

4.2 Testes e resultados:

Para teste foram adotados os seguintes valores como definição de comprimento do vetor x que será 10. Com isso fica automática determinada a espessura do obstáculo que é um décimo do comprimento no vetor x, que no caso será 1.

Porém as posições em que o obstáculo estiver será lido em um arquivo criado com o nome de "Dados.txt", vide os dados da tabela 4.1 e executado pelo código para cada uma das posições existentes no arquivo.

Estas posições foram valores atribuídos como exemplo, no arquivo a ser criado deve conter somente a segunda coluna, pois a primeira foi editada para melhor compreensão do valor adotado por interação.

interações	15
1	1.50
2	2.00
3	2.50
4	3.00
5	3.50
6	4.00
7	4.50
8	5.00
9	5.50
10	6.00
11	6.50
12	7.00
13	7.50
14	8.00
15	8.50


Total de interações

Tab. 4.1

O valor inicial indica quantos elementos existe no arquivo, já o segundo é o primeiro valor usado na interação. Os valores intermediários, a partir do segundo até o ultimo, possuem um fator de variação de 0.50.

Após o desenvolvimento do código, este criou para cada interação um arquivo determinando a posições dos nós, assim foi possível gerar

graficamente estes pontos como mostra respectivamente as figuras, sendo presumível a descrição do deslocamento como mostra as figuras: Fig. 4.2, Fig. 4.3, Fig. 4.4, Fig. 4.5, Fig. 4.6, Fig. 4.7, Fig. 4.8, Fig. 4.9, Fig. 4.10, Fig. 4.11, Fig. 4.12, Fig. 4.13, Fig. 4.14, Fig. 4.15, Fig. 4.16.

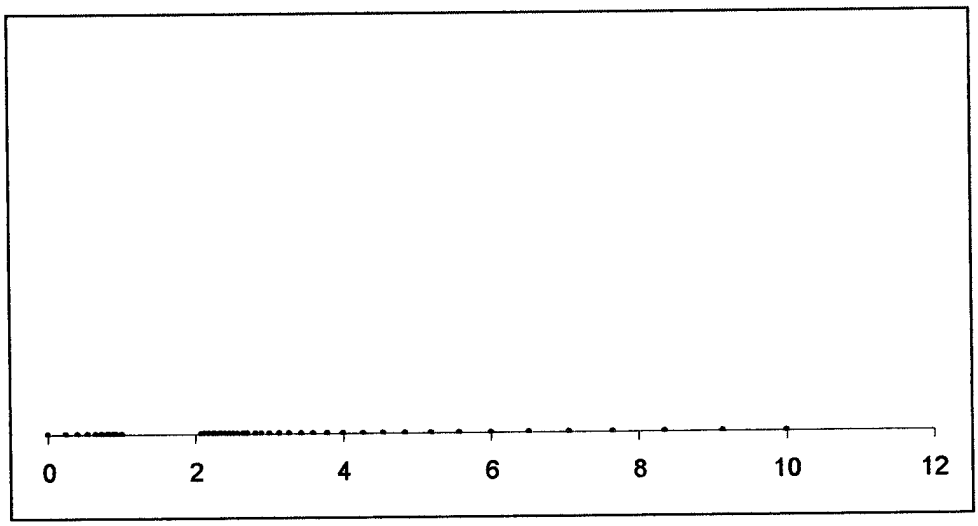

Fig. 4.2

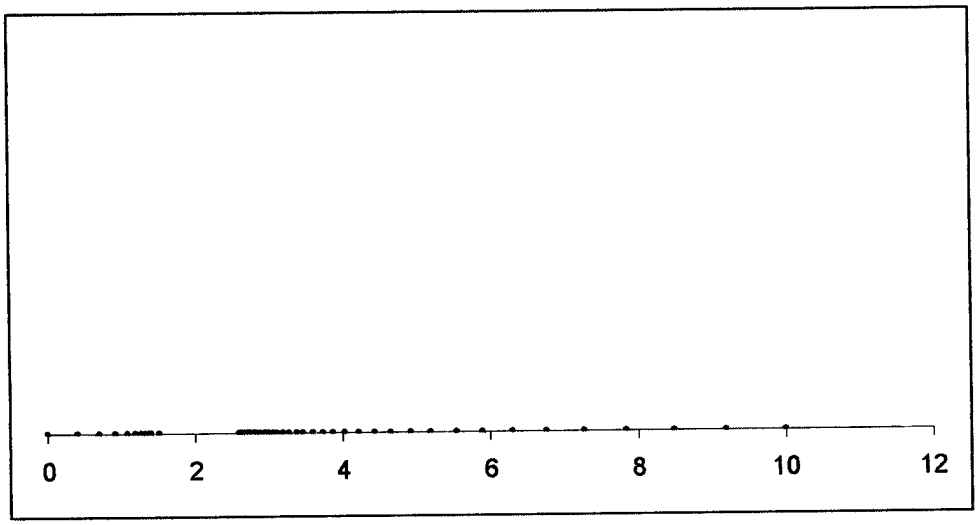

Fig. 4.3

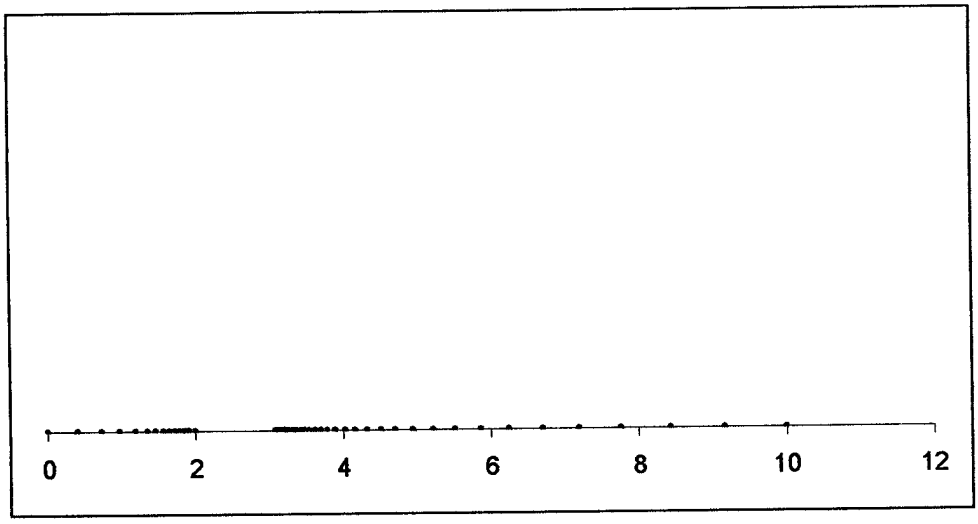

Fig. 4.4

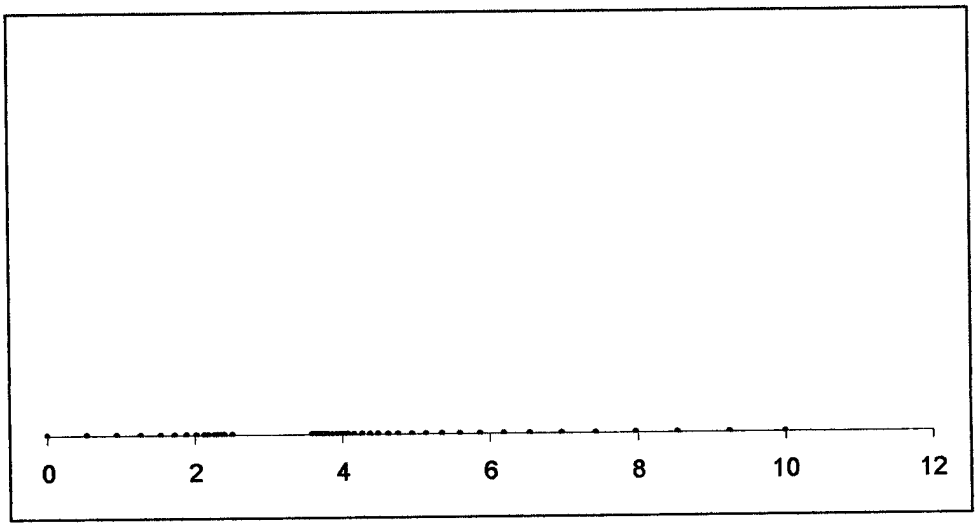

Fig. 4.5

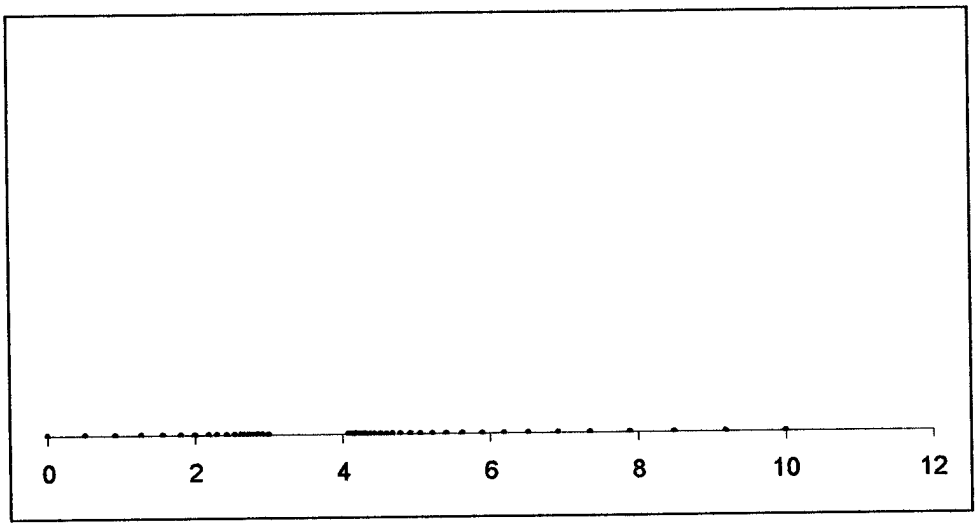

Fig. 4.6

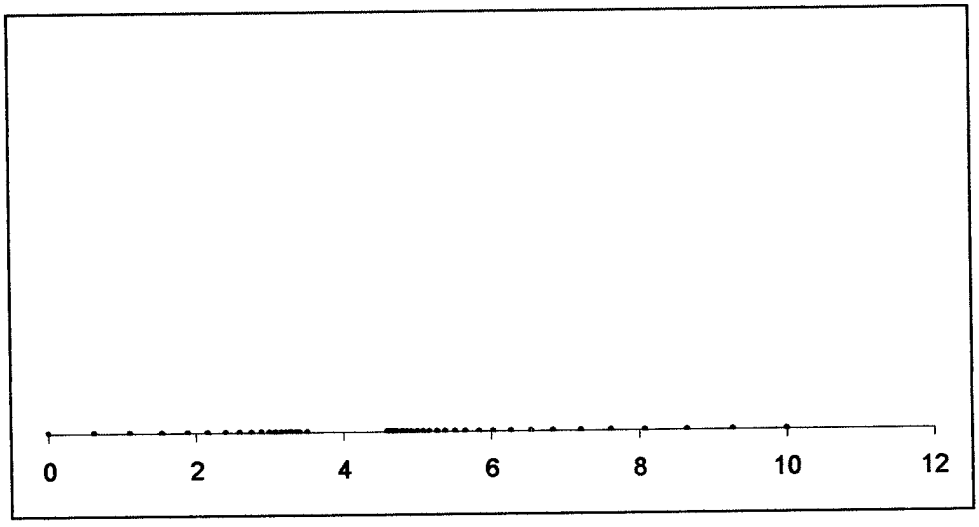

Fig.4.7

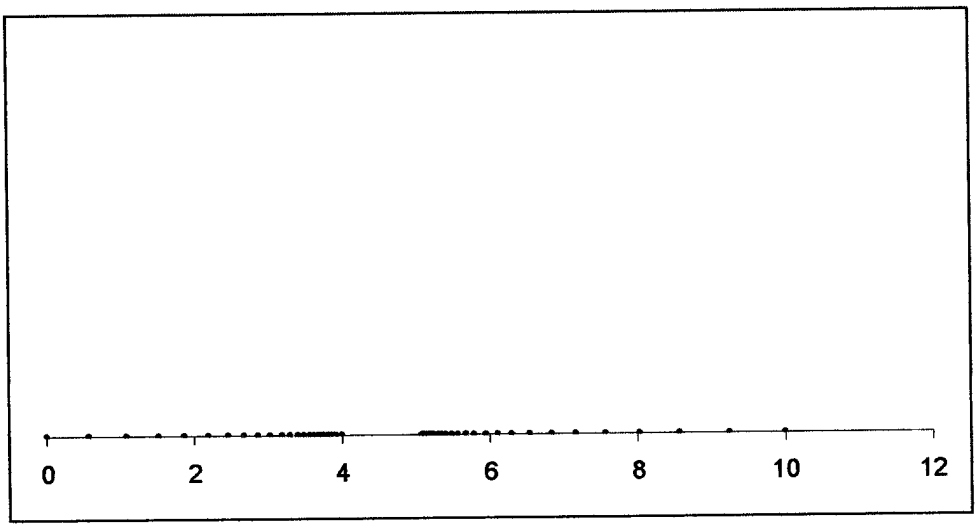

Fig. 4.8

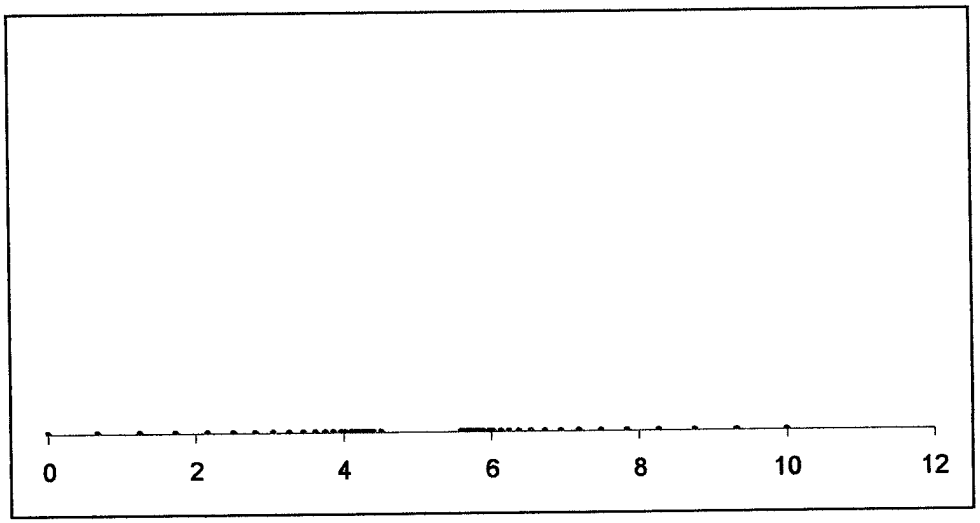

Fig. 4.9

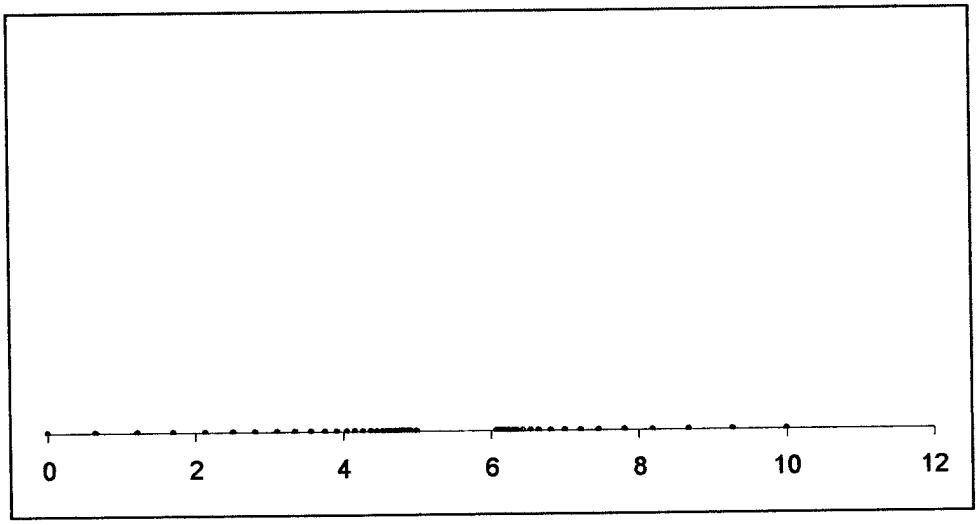

Fig. 4.10

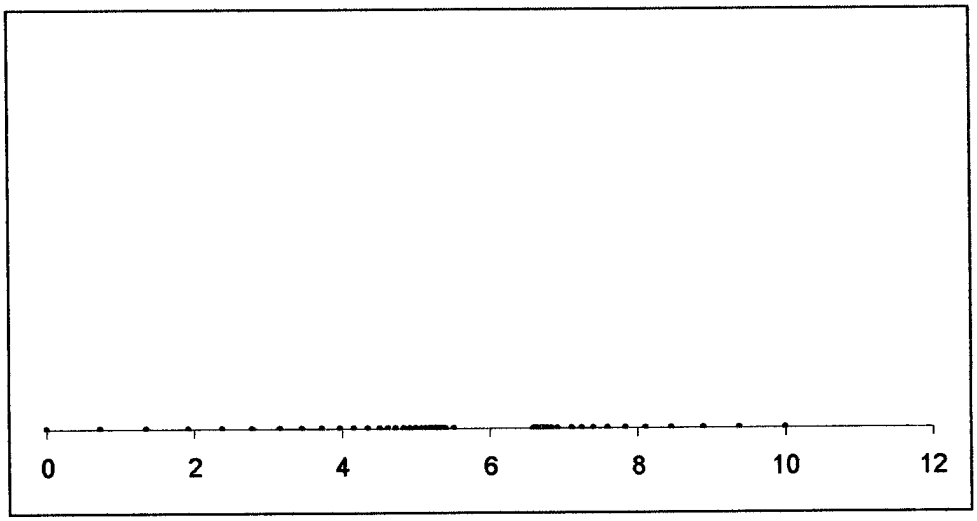

Fig. 4.11

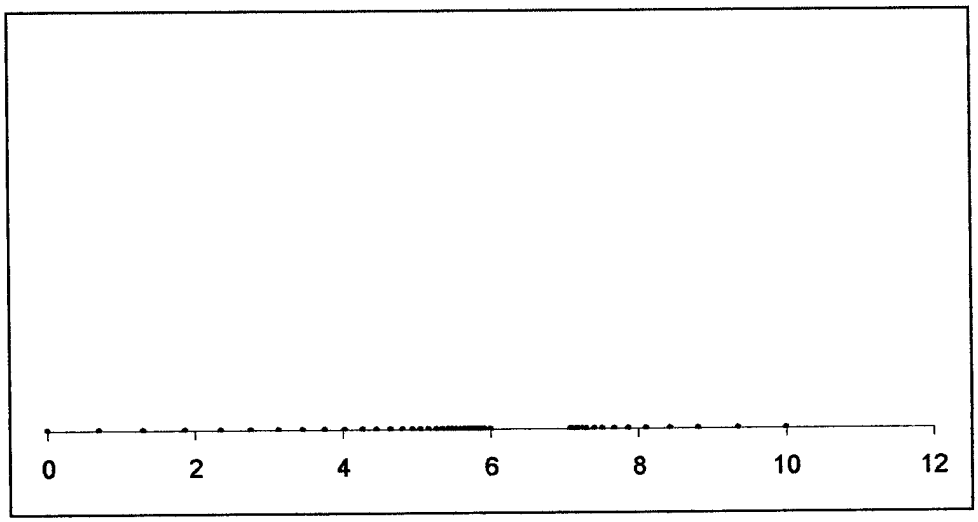

Fig. 4.12

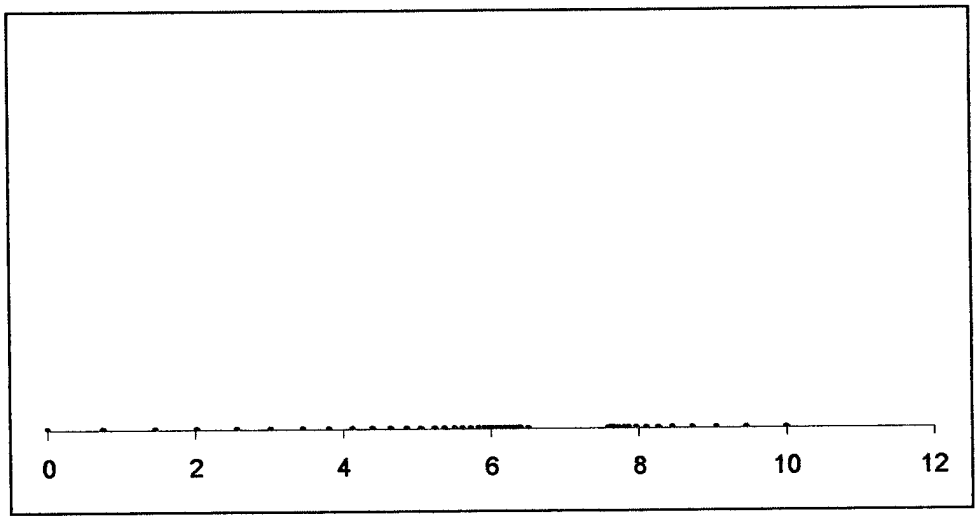

Fig. 4.13

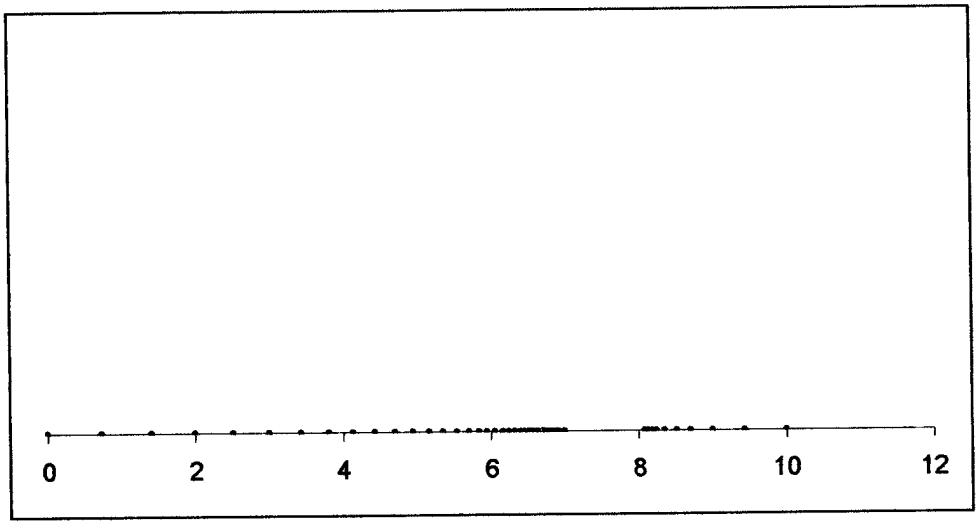

Fig. 4.14

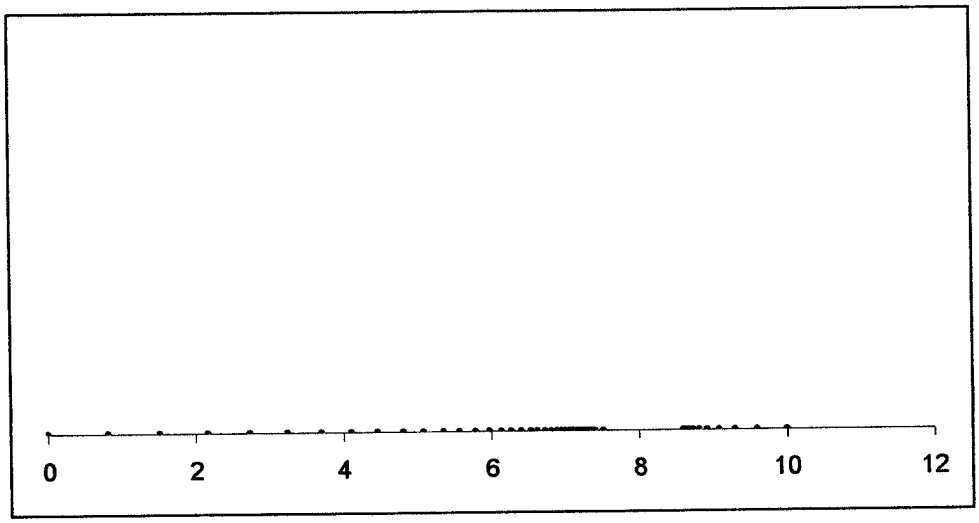

Fig. 4.15

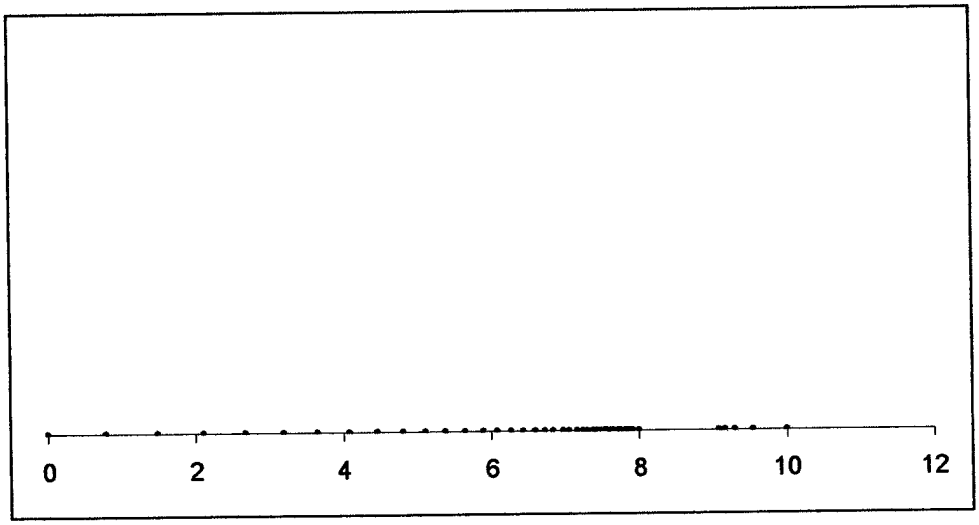

Fig. 4.16

5.1 Definições de cotas:

Para a construção da malha bidimensional são necessárias as definições das cotas, que são determinadas pelo usuário, de modo a definir com coerência as medidas a serem usadas são elas: comprimento (x), tamanho da redução do comprimento no vetor x na entrada, que será considerada igual na saída (k), altura no vetor y (y) e a posição do centro de gravidade do obstáculo no vetor x (x1).

Este último item porem deverá ser criado um arquivo com o nome "Dado.txt" para a inserção destas posições, de modo em que o primeiro número deve ser inteiro e positivo, pois determina o número de posições inserida no arquivo. Vide tabela 5.1.

No desenvolvimento do código fica automática determinada à espessura do obstáculo que será dado por um décimo do vetor x, estabelecido pelo usuário, a altura do obstáculo que é oito décimos da altura no vetor y, a redução do vetor y que é determinado como sendo um décimo da altura y, o centro de gravidade em y no ponto médio do vetor y.

Quando definido o tamanho do vetor x e o vetor y, definia se também o número de divisões em ambos vetores, mas foi criado para cada vetor um fator que multiplicará o mesmo, possibilitando assim uma maior divisão dos vetores, ou seja, um refinamento do intervalo dando assim uma melhor qualidade na analise dos gráficos. Estes fatores deverão ser determinados pelo usuário de forma coerente, devendo ser um numero inteiro e positivo.

No entanto foi desenvolvido um croqui para facilitar o entendimento das variáveis acima descritas, mostrando por meio da figura onde estará sendo aplicado o valor inserido pelo usuário.

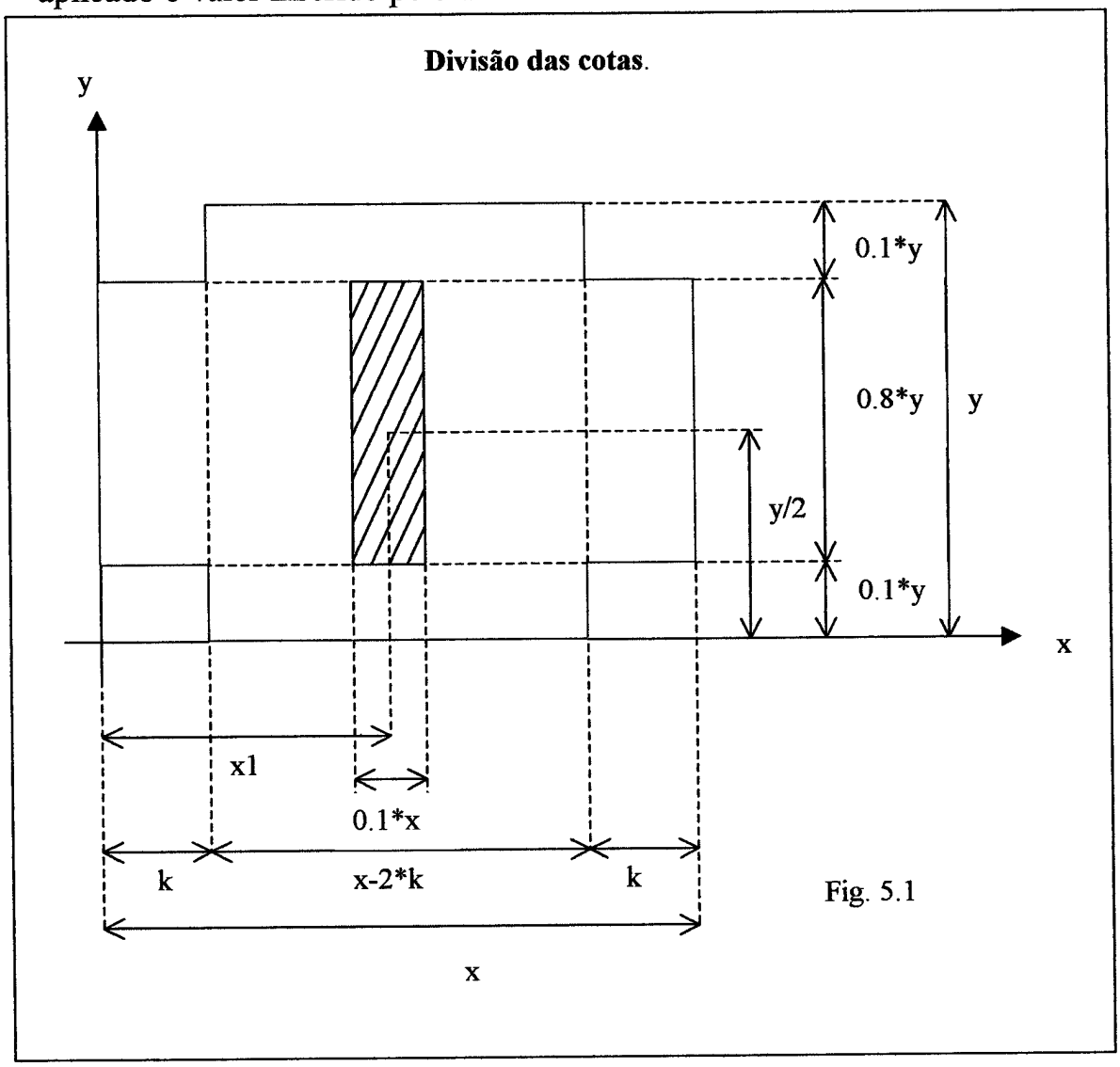

Fig. 5.1

O primeiro passo tomado pelo código é fazer a leitura do arquivo "Dado.txt", alocando um vetor com as posições desejadas, condicionando com que estes dados não seja menor e nem maior do que os números pertencentes ao domínio dos números reais possíveis de realizar o código com sucesso.

O posicionamento destes pontos no vetor x serão dados de forma progressiva, com intervalo decrescente, do ponto inicial até o obstáculo e de

forma progressiva, com intervalo crescente, do obstáculo até a parte final. No intervalo contido o obstáculo será dividido de forma homogênea com o numero de divisões determinado pelo usuário.

O posicionamento dos pontos no vetor y serão definidos de forma homogênea, com intervalo dado entre a ponto inicial até o ponto final.

Após o código evoluir todos estes passos os resultados obtidos serão salvos com o nome de "Coordenada_plotar_n. Xls". Onde o "n" será definido pelo número da interação, sendo o arquivo terá uma extensão do Excel, tais resultados serão possíveis plotar no Excel e visualizar estes pontos.

5.2 Testes e resultados:

Para teste foram adotados os seguintes valores como definição de comprimento do vetor x que será 10, para a altura no vetor y 10, para valor de corte 1. Com isso fica automática determinada a espessura do obstáculo que é um décimo do comprimento no vetor x que no caso será 1, valor da altura do obstáculo que é oito décimos da altura que no caso 8, corte no vetor y como sendo um décimo da altura que no caso 1 e a posição do centro de gravidade do obstáculo no vetor y que no caso 5.

Porém as posições em que o obstáculo estiver será lido em um arquivo criado com o nome de "Dados.txt", vide os dados da tabela 5.1 e executado pelo código para cada uma das posições existentes no arquivo.

Estas posições foram valores atribuídos como exemplo sendo que valor inicial indica quantos elementos existe no arquivo, já o segundo é o primeiro valor usado na interação. Os valores intermediários, a partir do segundo até o último, possuem um valor variação de 0.50.

No arquivo a ser criado deve conter somente a segunda coluna, pois a primeira foi editada para melhor compreensão do valor adotado por interação.

interações	15
1	1.50
2	2.00
3	2.50
4	3.00
5	3.50
6	4.00
7	4.50
8	5.00
9	5.50
10	6.00
11	6.50
12	7.00
13	7.50
14	8.00
15	8.50

⇦ Total de interações

Tab. 5.1

Para cada interação foi criado um arquivo determinando a posições dos nós com coordenada cartesiana (x,y), que foram obtidos graficamente no Excel como mostra respectivamente as figuras: Fig. 5.2, Fig. 5.3, Fig. 5.4, Fig. 5.5, Fig. 5.6, Fig. 5.7, Fig. 5.8, Fig. 5.9, Fig. 5.10, Fig. 5.11, Fig. 5.12, Fig. 5.13, Fig. 5.14, Fig. 5.15, Fig. 5.16.

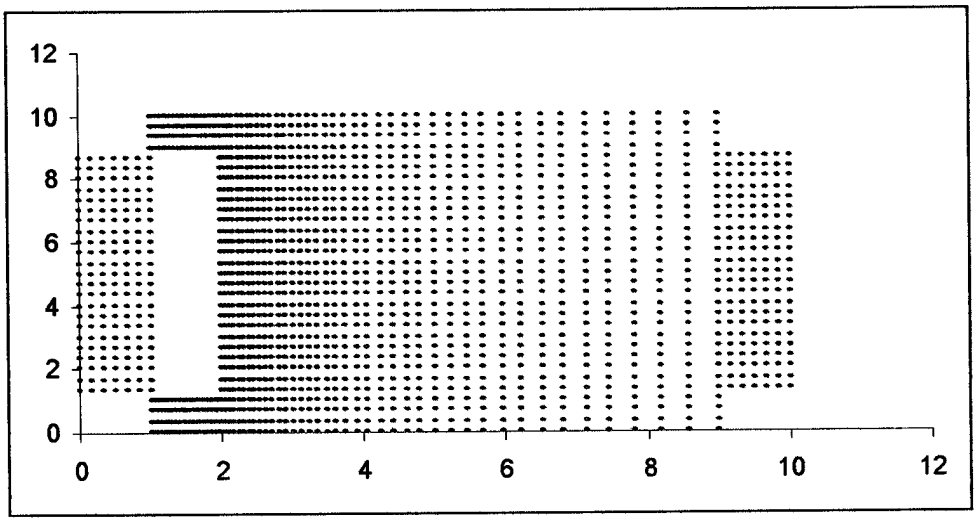

Fig. 5.2

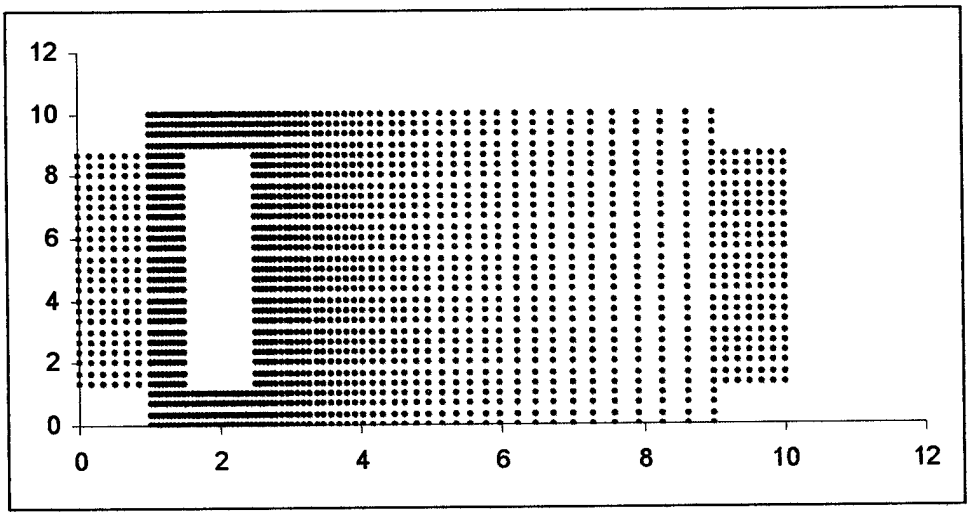

Fig. 5.3

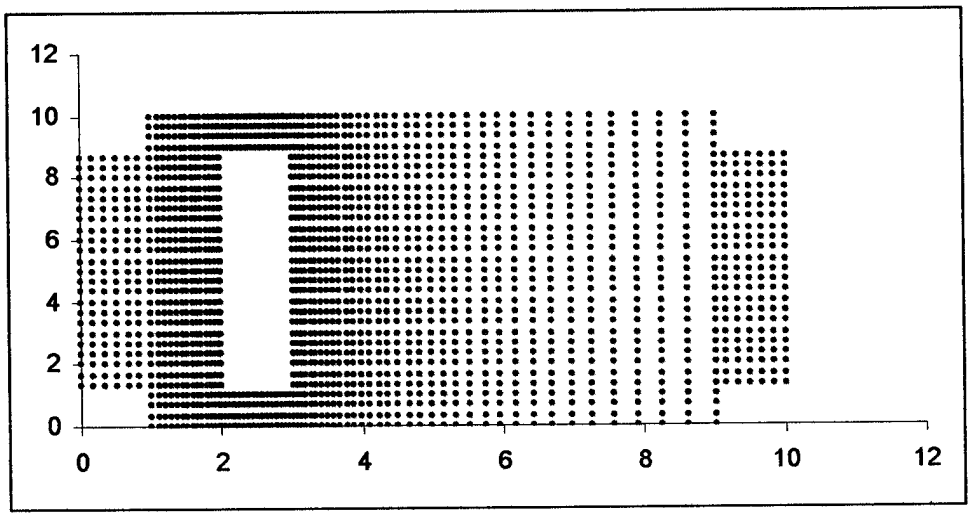

Fig. 5.4

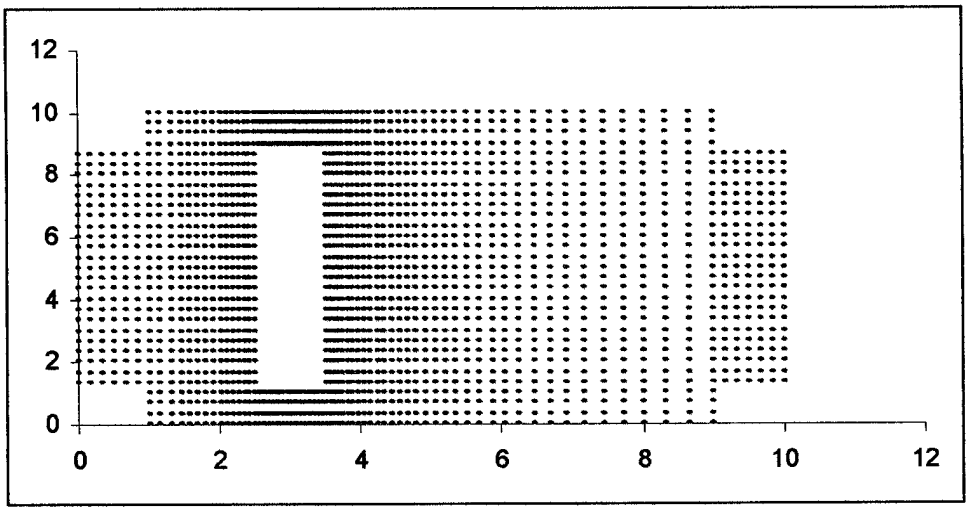

Fig. 5.5

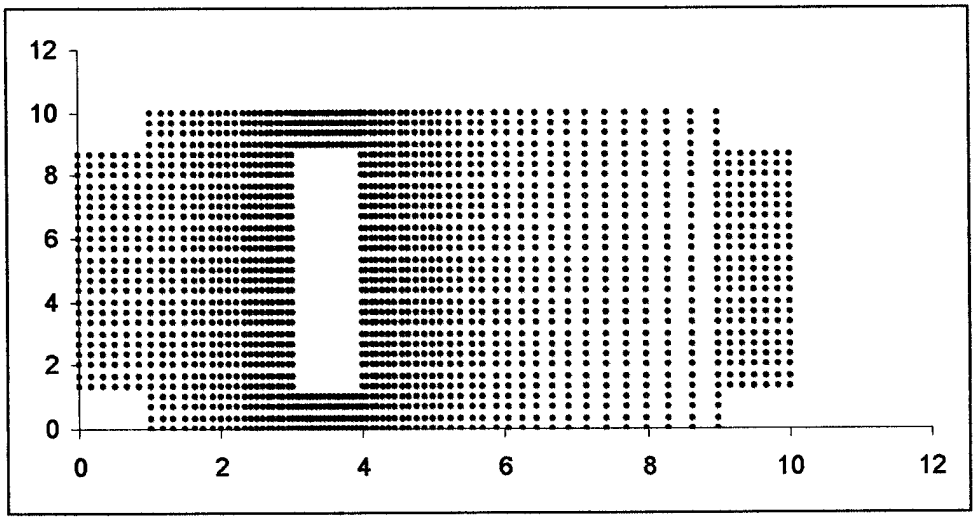

Fig. 5.6

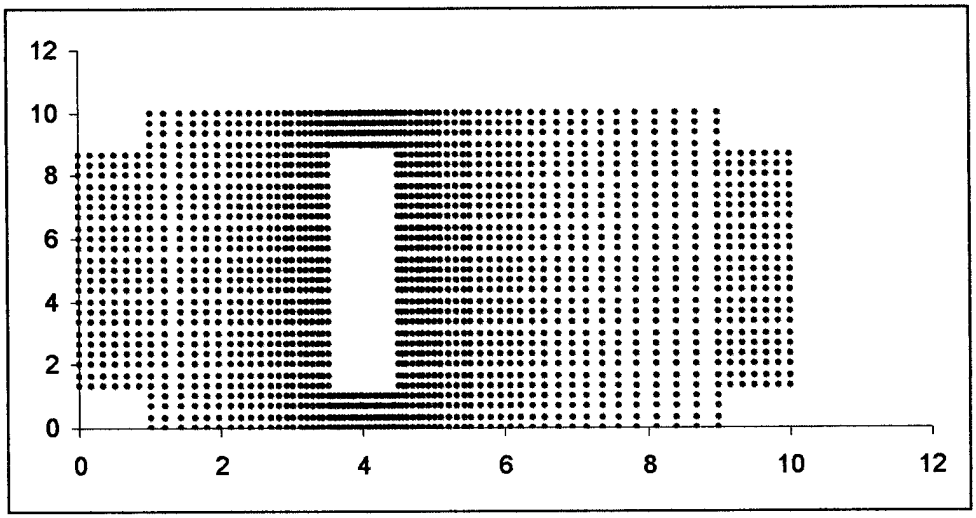

Fig. 5.7

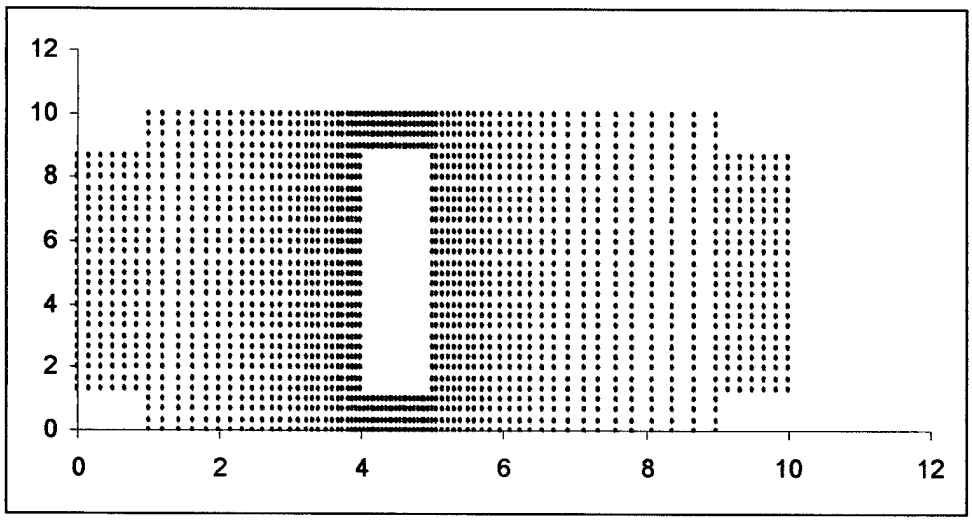

Fig. 5.8

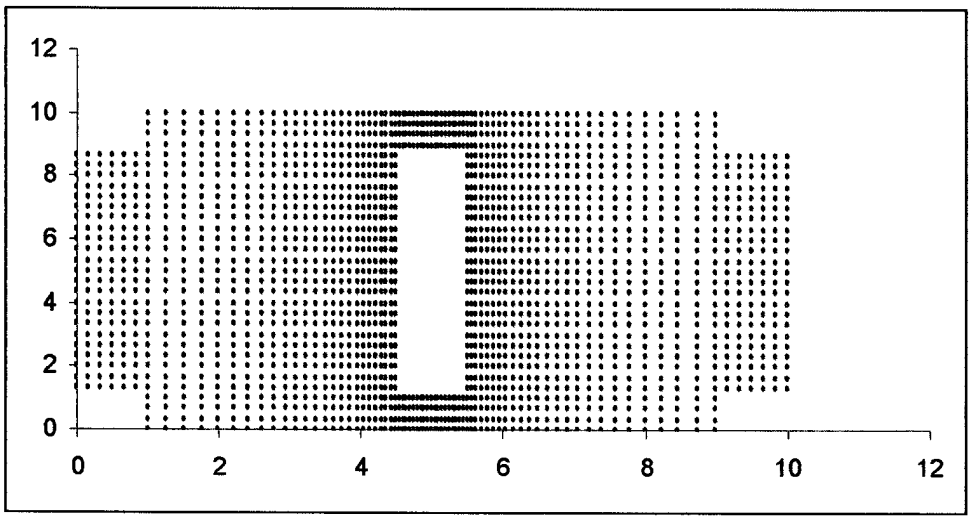

Fig. 5.9

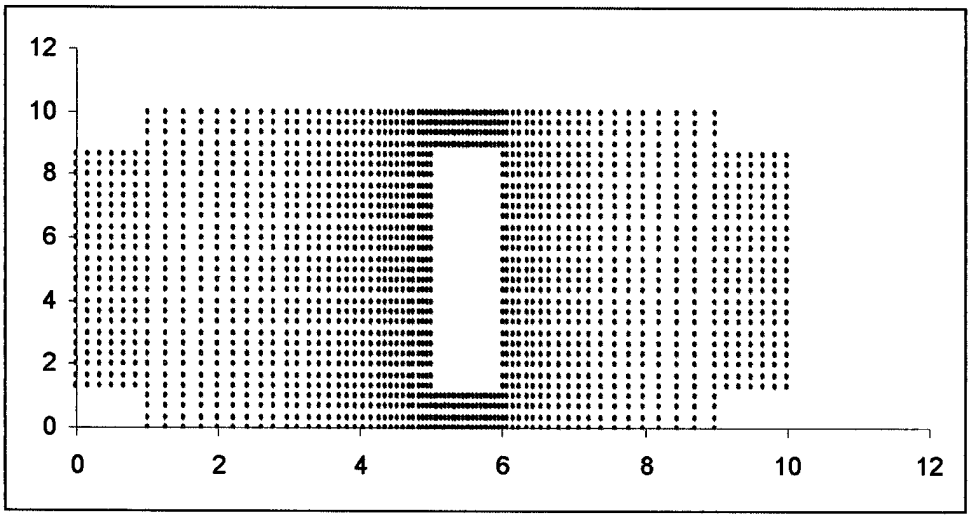

Fig. 5.10

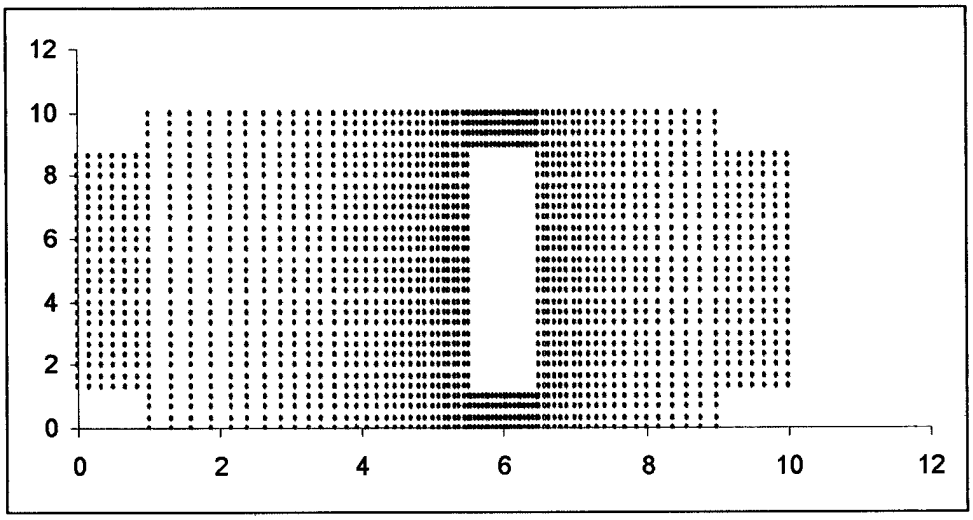

Fig. 5.11

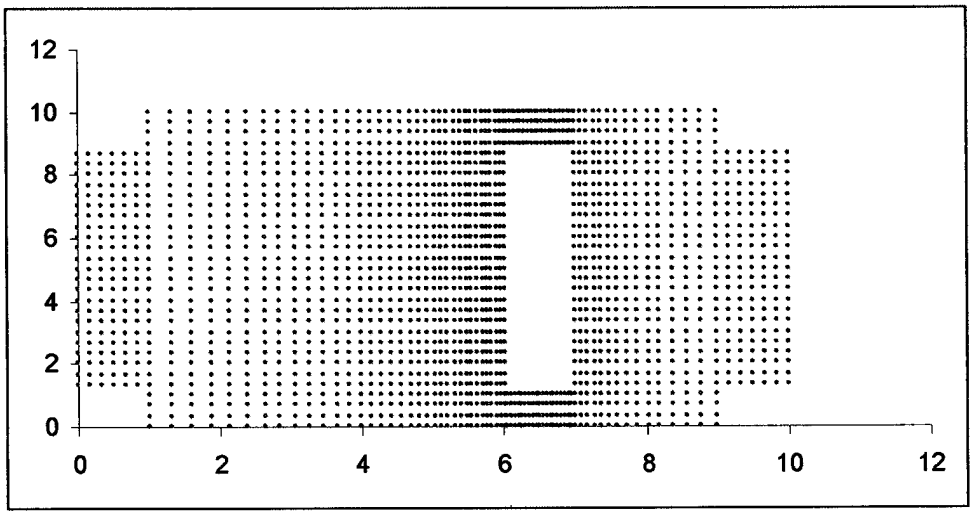

Fig. 5.12

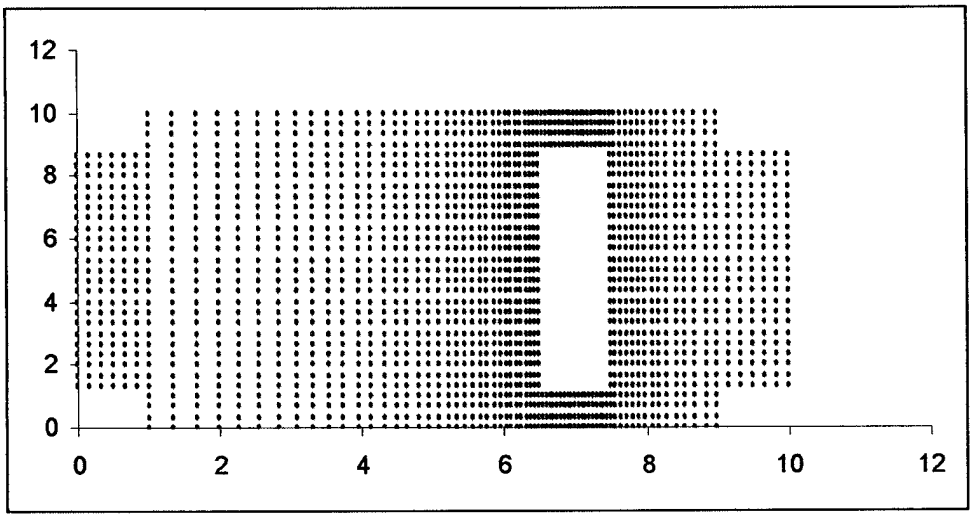

Fig. 5.13

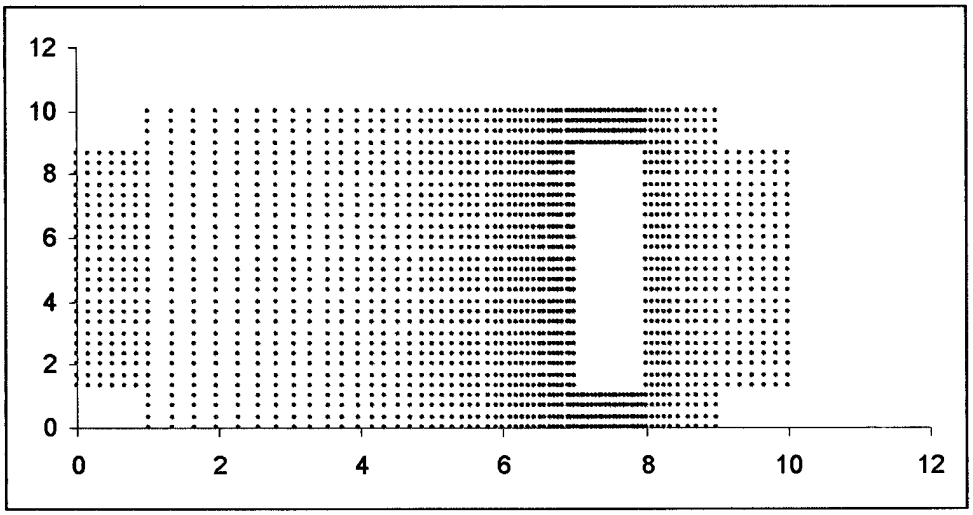

Fig. 5.14

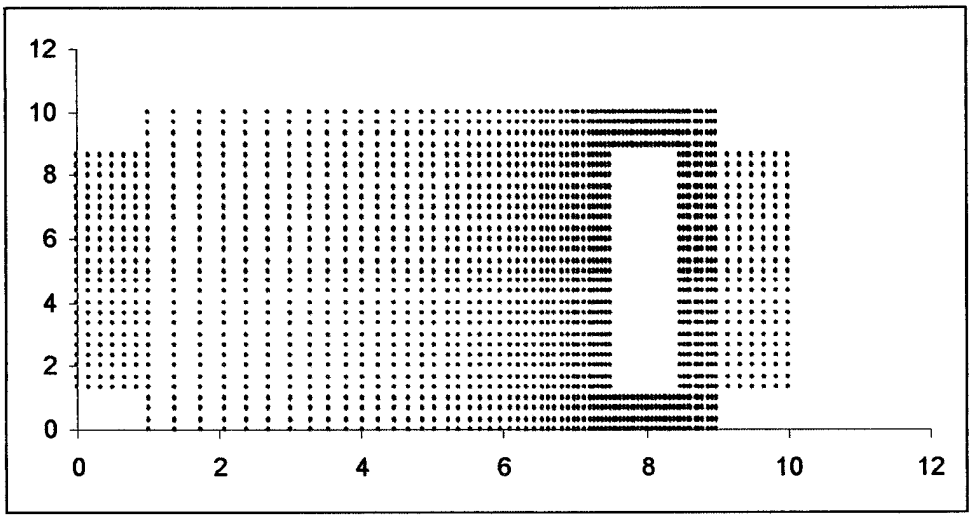

Fig. 5.15

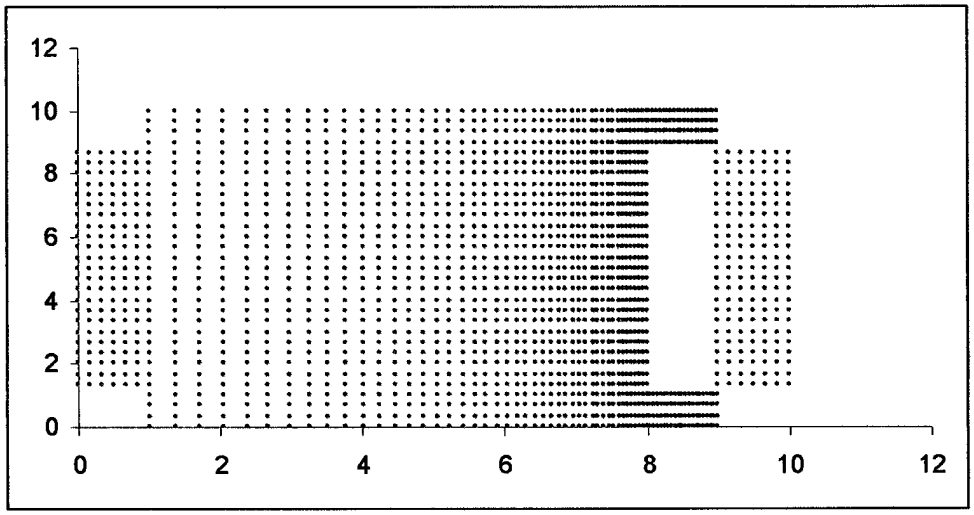

Fig. 5.16

6.1 Definições de cotas:

Para a construção da malha tridimensional são necessárias as definições das cotas em coordenada cilíndrica, que são determinadas pelo usuário, de modo a definir com coerência as medidas a serem usadas são elas: comprimento (z), tamanho do corte no vetor z (k), diâmetro (r), divisões na seção transversal, e a posição do centro de gravidade do obstáculo no vetor z (z1).

Este último item porem deverá ser criado um arquivo com o nome "Dado.txt" para a inserção destas posições, de modo em que o primeiro número deve ser inteiro e positivo determinando o numero de posições inserida no arquivo. Vide tabela 6.1.

No desenvolvimento do código fica automática determinada a espessura do obstáculo que será um décimo do comprimento no vetor z, este estabelecido pelo usuário, a altura do obstáculo será definido como sendo oito décimos da altura do diâmetro estabelecido, o corte no vetor z que é determinado como sendo um décimo do valor do diâmetro, o centro de gravidade no vetor z será o ponto médio do mesmo.

Quando definido o tamanho do diâmetro e o vetor z, definia se também com o mesmo número as divisões em ambos, mas foi criada para o diâmetro e o vetor z um fator que multiplicará o mesmo possibilitando assim uma maior divisão dos mesmos, ou seja, um refinamento do intervalo dando assim uma melhor qualidade na analise dos gráficos. Estes fatores deverão ser determinados pelo usuário de forma coerente, devendo ser um numero inteiro e positivo.

Também neste caso foi criado uma um fator que determina a divisão da circunferência frontal determinando a variação do angulo, alem da opção de definir o tamanho da aresta do cubo centrado no eixo z.

No entanto foi desenvolvido um croqui para facilitar o entendimento das variáveis acima descritas, mostrando por meio da figura onde estará sendo aplicado o valor inserido.

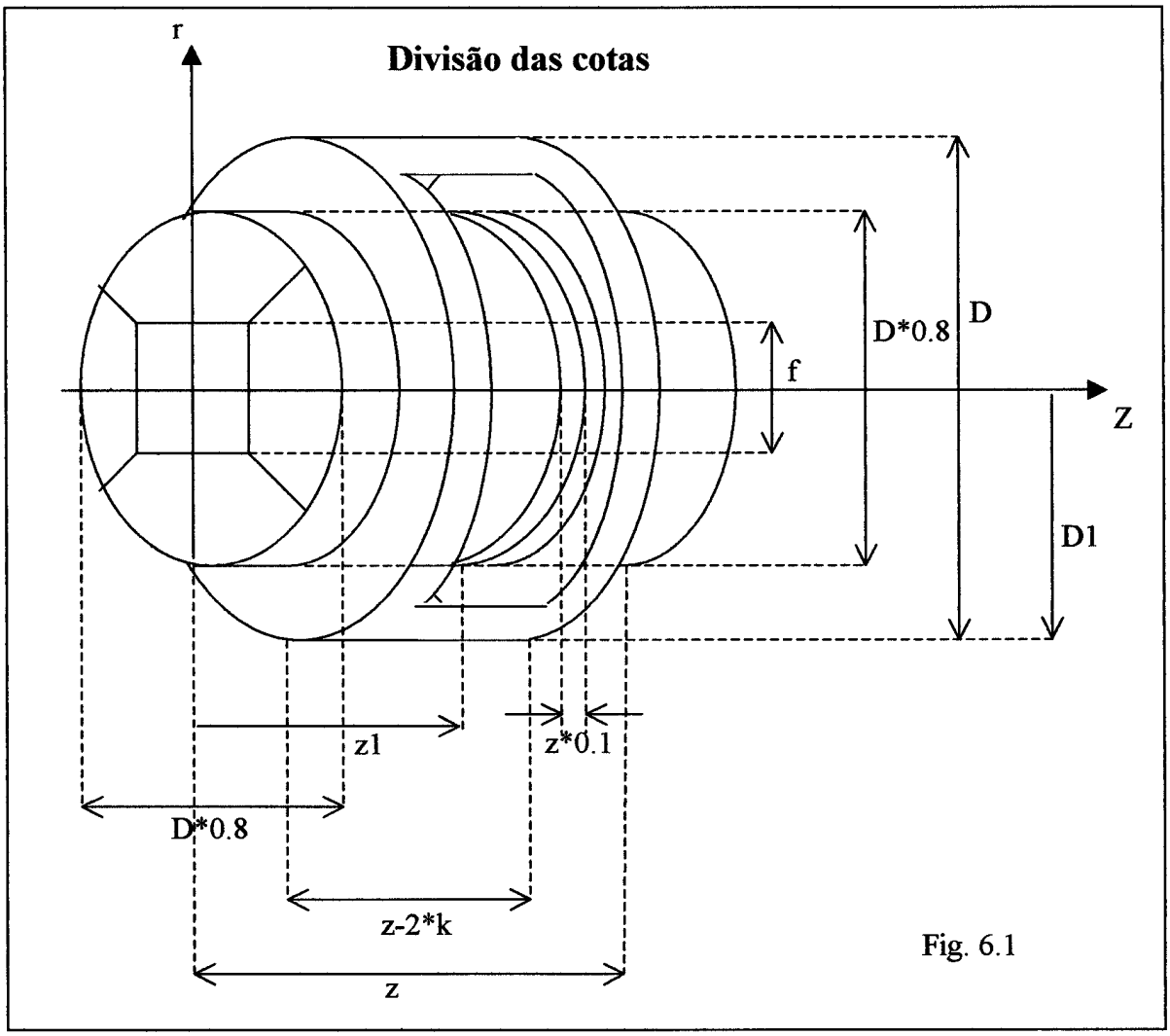

Fig. 6.1

O primeiro passo tomado pelo código é fazer a leitura do arquivo "Dado.txt", alocando um vetor com as posições desejadas, condicionando com

que estes dados não seja menor e nem maior do que os números pertencentes ao domínio dos números reais possíveis de realizar o código com sucesso.

O posicionamento destes pontos no vetor z serão dados de forma progressiva, com intervalo decrescente, do ponto inicial até o obstáculo e de forma progressiva, com intervalo crescente, do obstáculo até a parte final. No intervalo contido o obstáculo será dividido de forma homogênea com o numero de divisões determinado pelo usuário.

O posicionamento destes pontos no vetor r serão dados de forma homogênea, com intervalo da parte inicial até o fim de forma igualitária, além do número adotado na face deve ser inserido pelo usuário (f).

Após o código evoluir todos estes passos os resultados obtidos serão salvos com o nome de "Coordenada_plotar_n. Xls". Onde o "n" será definido pelo numero da interação, sendo um arquivo do Excel. Tais resultados serão possível plotar no Excel e visualizar estes pontos.

6.2 Testes e resultados:

Para teste foram adotados os seguintes valores como definição de comprimento do vetor z que será 10 (z), para o vetor r 10 (D), para valor de corte 1 (k). Com isso fica automática determinada a espessura do obstáculo que é um décimo do comprimento no vetor z que no caso será 1, valor da altura do obstáculo que é oito décimos do vetor r que no caso 8, corte no vetor z como sendo um decimo do vetor r que no caso 1, a posição do centro de gravidade do obstáculo no vetor r que no caso 5 (D1) e o numero adotado na aresta do cubo 1 (f).

Porém as posições em que o obstáculo estiver será lido em um arquivo criado com o nome de "Dados.txt", vide os dados da tabela 6.1 e executado pelo código para cada uma das posições existentes no arquivo.

Estas posições foram valores atribuídos como exemplo sendo que valor inicial indica quantos elementos existe no arquivo, já o segundo é o primeiro valor usado na interação. Os valores intermediários, a partir do segundo até o ultimo, possui um valor variação de 0.50.

No arquivo a ser criado deve conter somente a segunda coluna, pois a primeira foi editada para melhor compreensão do valor adotado por interação.

interações	15
1	1.50
2	2.00
3	2.50
4	3.00
5	3.50
6	4.00
7	4.50
8	5.00
9	5.50
10	6.00
11	6.50
12	7.00
13	7.50
14	8.00
15	8.50

⇦ Total de interações

Tab. 6.1

Devido ao desenvolvimento do código não foi possível gerar os resultados esperados, pois possue erro na edição do mesmo e fez com que atrasasse sua construção.

7.1 Resultados gerados:

Como trabalho adicional foi possível construir um modelo físico simples de modo a tomar a aceleração como constante e por meio matemático obter o deslocamento e velocidade.

Os resultados que foram obtidos no modelo físico esta apresentada de forma gráfica de modo a poder de identificar as projeções tomadas pêlos pontos resultantes.

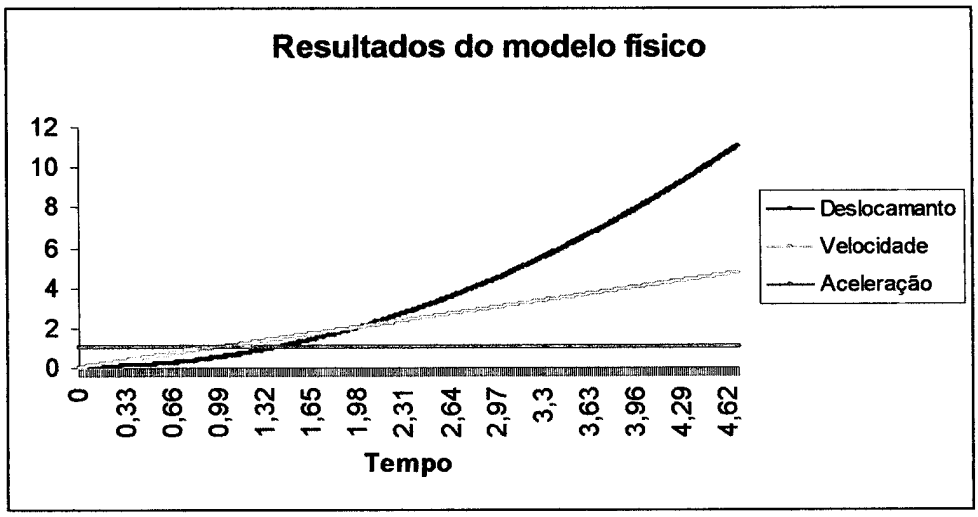

Fig. 7.1

CAP. 8 MODELO COMPUTACIONAL

Neste capítulo será dada ênfase na estrutura computacional do programa que foi desenvolvido para a simulação numérica de formação da malha para os casos unidimensional, bidimensional e tridimensional.

8.1 Organização do Programa:

O modo com são organizadas as rotinas é de fundamental importância na abstração do problema. Os módulos devem ser organizados de acordo com o que se deseja futuramente do programa, considerando suas modificações, atualizações, etc. Neste trabalho, a organização do programa foi feita de modo que cada etapa do processo de solução, dada no modelo matemático, fosse feita em um módulo independente. Ou seja, caso precise-se modificar alguma rotina em algum módulo, não é necessário influir em outras partes do código. Isto é necessária principalmente nas etapas de geração da geometria e posteriormente uma adaptação do cálculo do escoamento.

8.2 Construção do Código:

O programa foi escrito em linguagem C, usufruindo as ferramentas da programação orientada a objetos, como a construção de classes e objetos, mas mantendo a clareza e facilidade.

8.3 Código 1D:

```
//_____________________________________________________________________
//| PROGRAMA INSTITUCIONALDE DE BOLSA DE INICIAÇÃO CIENTIFICA .       |
//|                                                                   |
//|                        GERADOR DE MALHA 1D.                       |
//|          (Programa que tem por finalidade definir os pontos da malha.)  |
//|                                                                   |
//| Autor: JORGE LUIZ DEMETRIO.                                       |
//| Orientador: JERONIMO S. TRAVELHO.                                 |
//|                                                                   |
//| Data: Abril/2004.                                                 |
//|                                                                   |
//| Compilador: CC Linguagem C.                                       |
//|___________________________________________________________________|

//_____________________________________________________________________
//                  Incluindo biblioteca.
//_____________________________________________________________________
#include <stdio.h>
#include <malloc.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>
//_____________________________________________________________________

//_____________________________________________________________________
//                    Declaração global das variáveis.
//_____________________________________________________________________
char op;
float *vetorx,*posicao;
int i,j,xri,n,cont;
float x1a,x1b,x,k;
//_____________________________________________________________________

void Coordenadax (void)
  {
//_____________________________________________________________________
//                    Declaração local de variáveis.
//_____________________________________________________________________
  char indice[40],chartemp[40],auxiliar[40];
  int fatr,divr,ia,ib,x1r,u;
  float x1,xr,fat,div,tam,ce1,ce2,an,a1,q,Aux;
  FILE *entr,*saida;
//_____________________________________________________________________
  printf("\nDigite o valor do comprimento do vetor x.\n");
```

```
scanf("%f",&x);
do
 {
 printf("\nDigite o fator que multiplicara o numero pon-\n");
 printf("tos do vetor x.\n");
 scanf("%f",&fat);
 fatr = floor(fat);
 if (fatr!=fat)
  {
  printf("\nO valor do fator deve ser um valor inteiro\n");
  printf(", caso resposta for nao sera adotado 1..\n");
  printf("\nDeseja tentar novamente(S,N)?\n");
  scanf("%c",&op);
  op=(char)getchar();
  fatr=1;
  }
  else
   op='N';
 }
while(op=='S');

do
 {
 printf("\nDigite o numero de divisoes do intervalo onde\n");
 printf("esta contido o obstaculo.\n");
 scanf("%f",&div);
 div r= floor(div);
 if (divr!=div)
  {
  printf("\nO valor da divisoes deve ser um valor inteiro\n");
  printf(", caso resposta for nao sera adotado 0..\n");
  printf("\nDeseja tentar novamente(S,N)?\n");
  scanf("%c",&op);
  op=(char)getchar();
  fatr=0;
  }
  else
   op='N';
 }
while (op=='S');

xr=x*fatr+div+1;
xri=(int)floor(xr);
vetorx = (float *) malloc (xri*sizeof (float));
k=0;

for (i=0;i<xri;i++)
```

```
  vetorx[i]=k;

printf("==========INFORMACAO========== \n");
printf("|     O tamanho do obstaculo sera estabelecido co-      | \n");
printf("|     mo sendo 1/10 do valor comprimento.               | \n");
printf("============================== \n");

tam=(float)x*0.1;
cont=0;
do
  {
  entr=fopen("Dado.txt","r");
  fscanf(entr,"%d",&u);
  posicao = (float*) malloc (u*sizeof(float) );
  for (j=0;j<u;j++)
    fscanf (entr,"%f",&posicao[j]);

  for (i=0;i<u;i++)
    printf ("%7.2f\n",posicao[i]);

  x1=0;
  while(x1!=posicao[u-1])
    {
    cont++;
    x1=posicao[cont];

    ce1=tam/2;
    ce2=x-(tam/2);
    x1r=(int)floor(x1+.5);

    if (x1>=ce1 && x1<=ce2)
      {
      x1a=(float)x1-(x*0.1)/2;
      ia=(int)floor(x1a+.5);
      x1b=(float)x1+(x*0.1)/2;
      ib=(int)floor(x1b+.5);

      printf("          Dados          \n");
      printf(" ---------------------- \n");
      printf("|Arredondamento = %d | \n",x1r);
      printf("|x1b = %5.2f           | \n",x1b);
      printf("|x1a = %5.2f           | \n",x1a);
      printf("|ia = %d               | \n",ia);
      printf("|ib = %d               | \n",ib);
      printf(" ------------------ ------- \n");

//_____________________________________________________________
```

```
//              Definindo divisões do vetor x.
//_____________________________________________________________________
      vetorx[0]=x1a;
      n=1;

      if(x1r>x1)
       ia=ia+1;

      if(x1>tam/2)
       {
       while (n<=ia*fat)
        {
        an=x1a;
        a1=0.09;
        q=pow((an/a1),(1/(x1r*fat-1)));
        vetorx[n]=x1a-a1*pow(q,(n-1));
        n++;
        }
       }

      q=0;
      while (n<=div+ia*fat)
       {
       q=q+tam/div;
       vetorx[n]=x1a+q;
       n++;
       }
      k=1;
      while (n<xri)
       {
       an=x-x1b;
       a1=0.09;
       q=pow((an/a1),(1/((x-x1r)*fat-1)));
       vetorx[n]=x1b+a1*pow(q,(k-1));
       n++;
       k++;
       }

      printf("\nn = %d\n",n);
      printf("\nOrganizando dados...\n");

      for (j=0;j<xri;j++)
       for (i=0;i<xri-1;i++)
        {
        if (vetorx[i]>=vetorx[i+1])
         {
         Aux=vetorx[i+1];
```

```
          vetorx[i+1]=vetorx[i];
          vetorx[i]=Aux;
          }
        }
//_____________________________________________________________________

      op='N';
      }
      else
       {
       if(x1<=ce1)
        {
        printf("================INFORMACAO=================== \n");
        printf("|          O valor inserido nao possibilita o desenvol-          | \n");
        printf("|       vimento da malha, menor valor possivel para a            | \n");
        printf("|       localizacao e %5.2f.                                      | \n",ce1);
        printf("=============================================== \n");
        }
       if(x1>=ce2)
        {
        printf("================INFORMACAO=================== \n");
        printf("|          O valor inserido nao possibilita o desenvol-          | \n");
        printf("|       vimento da malha, maior valor possivel para a            | \n");
        printf("|       localizacao e %5.2f.                                      | \n",ce2);
        printf("=============================================== \n");
        }
       printf("Deseja tentar novamente(S,N)?\n");
       scanf("%c",&op);
       op=(char)getchar();
       }

    printf("================INFORMACAO=================== \n");
    printf("|          O obstaculo esta compreendido entre o inter-          | \n");
    printf("|       valo de:                                                 | \n");
    printf("|          %7.2f <= OBSTACULO <= %7.2f                           | \n",x1a,x1b);
    printf("=============================================== \n");

//_____________________________________________________________________
//    Imprimindo na tela os resultados obtido das divisões do vetor x.
//_____________________________________________________________________
    printf("================INFORMACAO=================== \n");
    printf("       Identificando os nos que se apresentam na ma-           \n");
    printf("     "lha.                                                      \n");
    printf("=============================================== \n");
for (i=0;i<xri;i++)
     printf ("%7.2f\n",vetorx[i]);
```

```
    printf("=============================================================== \n");
//_____________________________________________________________________________

    printf("\n============================================================== \n");
    printf("                          RESUTADOS                             \n");
    printf("          Coordenada dos pontos contido na malha                \n");
    printf("============================================================== \n");

//_____________________________________________________________________________
//    Executando loop no nome do arquivo para armazenamento de dados.
//_____________________________________________________________________________
    strcpy(chartemp,"Coordenada_plotar_");
    strcpy(auxiliar,chartemp);

    sprintf (indice,"%d",cont);
    strcat (chartemp,indice);
    strcat (chartemp,".xls");
    printf("%s\n",chartemp);
    saida = fopen (chartemp,"w");
    strcpy(chartemp,auxiliar);
//_____________________________________________________________________________

    for(i=0;i<xri;i++)
      {
      getchar();
      fprintf(saida,"%5.2f\n",vetorx[i]);
      printf("%5.2f",vetorx[i]);
      }

    fclose(saida);

    printf("============================================================== \n\n");
    }

   fclose(entr);
   }
  while(op=='S');

  }

main ()
  {
  printf("\nInicializando...\n\n");
  printf(" ____________________________________________________________ \n");
  printf("|                    GERADOR DE MALHA 1D.                      |\n");
  printf("|      (Programa que tem por finalidade definir os pontos da malha.)  |\n");
  printf("|                                                              |\n");
```

```
 printf("|  Autor: JORGE LUIZ DEMETRIO.                                  |\n");
 printf("|  Orientador: JERONIMO S. TRAVELHO.                            |\n");
 printf("|                                                               |\n");
 printf("|  Data: Abril/2004.                                            |\n");
 printf("|                                                               |\n");
 printf("|  Compilador: CC Linguagem C.                                  |\n");
printf("|_______________________________________________________________|\n");
 printf("\n");

 Coordenadax ();

 printf("\nAtualizando arquivo...\n\n");

 printf("\nFinalizando.\n\n");
 return(0);
 }
```

8.4 Código 2D:

```
//_____________________________________________________________________
//| PROGRAMA INSTITUCIONALDE DE BOLSA DE INICIAÇÃO CIENTIFICA          |
//|                                                                    |
//|                          GERADOR DE MALHA 2D.                      |
//|          (Programa que tem por finalidade definir os pontos da malha.)|
//|                                                                    |
//| Autor: JORGE LUIZ DEMETRIO.                                        |
//| Orientador: JERONIMO S. TRAVELHO.                                  |
//|                                                                    |
//| Data: Abril/2004.                                                  |
//|                                                                    |
//| Compilador: CC Linguagem C.                                        |
//|____________________________________________________________________|

//____________________________________________________________________
//                        Incluindo biblioteca.
//____________________________________________________________________
#include <stdio.h>
#include <malloc.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>
//____________________________________________________________________

main ()
 {

//____________________________________________________________________
//                        Declaração das variáveis.
//____________________________________________________________________

 char indice[40],chartemp[40],auxiliar[40],op;
  float *vetor1x,*vetor2x,*vetor3x,*vetorEx,*vetorSx;
  float *vetor1y,*vetor2y,*vetor3y,*posicao;
  int i,j,xr,yr,iax,ibx,iay,iby,n,al,al1,al2,u,cont;
  float x1a,x1b,y1a,y1b,fat,faty,divi,dESf,k,tam,x,y;
  int fatr,divr,dESi,x1r,fatry,y1r;


  float x1,ce1,ce2,an,a1,q,Aux,y1,tamy;
  FILE *saida;
  FILE *entr;
//____________________________________________________________________
```

```
printf("\nInicializando...\n\n");

printf(" _____ \n");
printf("|                    GERADOR DE MALHA 2D.                    |\n");
printf("|        (Programa que tem por finalidade definir os pontos da malha.)        |\n");
printf("|                                                            |\n");
printf("|  Autor: JORGE LUIZ DEMETRIO.                               |\n");
printf("|  Orientador: JERONIMO S. TRAVELHO.                         |\n");
printf("|                                                            |\n");
printf("|  Data: Abril/2004.                                         |\n");
printf("|                                                            |\n");
printf("|  Compilador: CC Linguagem C.                               |\n");
printf("|____________________________________________________________|\n");
printf("\n");

printf("\nDigite o valor do comprimento do vetor x.\n");
scanf("%f",&x);
//_____________________________________________________________
//          Valor atribuido ao comprimento do vetor x.
//_____________________________________________________________
// x=10;
//_____________________________________________________________

xr=(int)floor(x+.5);
do
  {
  printf("\nDigite o fator que multiplicara o numero pon-\n");
  printf("tos do vetor x.\n");
  scanf("%f",&fat);
  fatr=(int)floor(fat);
  if (fatr!=fat)
    {
    printf("\nO valor do fator deve ser um valor inteiro\n");
    printf(", caso resposta for nao sera adotado 1.\n");
    printf("\nDeseja tentar novamente(S,N)?\n");
    scanf("%c",&op);
    op=(char)getchar();
    fatr=1;
    }
    else
      op='N';
  }
 while(op=='S');

//_____________________________________________________________
//          Valor atribuído ao fator que multiplicara o vetor x.
//_____________________________________________________________
```

```
// fat=5;
// fatr=5;
//_____________________________________________________________________

 do
  {
  printf("\nDigite o numero de divisoes do intervalo onde\n");
  printf("esta contido o obstaculo.\n");
  scanf("%f",&divi);
  divr=(int)floor(divi);
  if (divr!=divi)
   {
   printf("\nO valor da divisoes deve ser um valor inteiro\n");
   printf(", caso resposta for nao sera adotado 0.\n");
   printf("\nDeseja tentar novamente(S,N)?\n");
   scanf("%c",&op);
   op=(char)getchar();
   divr=0;
   }
   else
    op='N';
  }
 while (op=='S');

//_____________________________________________________________________
//              Numero atribuído a quantidade de divisoes no intervalo
//                        em que o obstáculo esta contido.
//_____________________________________________________________________
// divi=20;
// divr=20;
//_____________________________________________________________________
 do
  {
  printf("\nDigite o numero  de  divisoes do intervalo da\n");
  printf("entrada e saida.\n");
  scanf("%f",&dESf);
  dESi=(int)floor(dESf);
  if (dESi!=dESf)
   {
   printf("\nO valor da divisoes deve ser um valor inteiro\n");
   printf(", caso resposta for nao sera adotado 1.\n");
   printf("\nDeseja tentar novamente(S,N)?\n");
   scanf("%c",&op);
   op=(char)getchar();
   dESi=1;
   }
   else
```

```
      op='N';
    }
  while (op=='S');

//_____________________________________________________________________
//      Numero atribuído quantidade de divisoes na entrada e saída.
//_____________________________________________________________________
// dESi=4;
// dESf=4;
//_____________________________________________________________________

  printf("\nDeterminar o tamanho do corte no vetor x.\n");
  scanf("%f",&a1);

//_____________________________________________________________________
//      Numero atribuido ao corte da valvula.
//_____________________________________________________________________
// a1=1;
//_____________________________________________________________________

  printf("\nDigite o valor do comprimento do vetor y.\n");
  scanf("%f",&y);

//_____________________________________________________________________
//       Valor atribuido a altura do vetor y.
//_____________________________________________________________________
// y=10;
//_____________________________________________________________________

  do
    {
    printf("\nDigite o fator que multiplicara o numero pon-\n");
    printf("tos do vetor y.\n");
    scanf("%f",&faty);
    faty=3;
    fatry=floor(faty);
    if (fatry!=faty)
      {
      printf("\nO valor do fator deve ser um valor inteiro\n");
      printf(", caso resposta for nao sera adotado 1.\n");
      printf("\nDeseja tentar novamente(S,N)?\n");
      scanf("%c",&op);
      op=(char)getchar();
      fatry=1;
      }
      else
        op='N';
```

```
  }
 while(op=='S');

//_____________________________________________________________________
//        Valor atribuído ao fator que multiplicara o vetor y.
//_____________________________________________________________________
// faty=5;
// fatry=5;
//_____________________________________________________________________

 printf("================INFORMACAO================== \n");
 printf("|          O tamanho do obstaculo sera estabelecido co-          |\n");
 printf("|       mo sendo 1/10 do valor comprimento.                      |\n");
 printf("============================================== \n");
 tam=x*0.1;

 do
  {
  entr=fopen("Dado.txt","r");
  fscanf(entr,"%d",&u);
  posicao = (float*) malloc (u*sizeof (float) );
  for (j=0;j<u;j++)
   fscanf (entr,"%f",&posicao[j]);

  x1=0;
  cont=0;
  while(x1!=posicao[u-1])
   {
   x1=posicao[cont];
   cont++;
   ce1=(tam/2)+a1;
   ce2=(x-(tam/2)-a1);
   x1r=(int)floor(x1+.5);

   if (x1>=ce1 && x1<=ce2)
    {
    x1a=(float)x1-(x*0.1)/2;
    iax=(int)floor(x1a+.5);
    x1b=(float)x1+(x*0.1)/2;
    ibx=(int)floor(x1b-.5);

    if(x1r>x1)
     {
     iax=iax+1;
     ibx=ibx+1;
     }
```

```
vetor1x = (float *) malloc (iax*fat*sizeof (float));
vetor2x = (float *) malloc ((divi*sizeof (float))-1);
vetor3x = (float *) malloc ((xr-ibx)*fat*sizeof (float));
vetorEx = (float *) malloc (dESf*sizeof (float));
vetorSx = (float *) malloc (dESf*sizeof (float));

printf("\n  Dados        \n");
printf(" ---------------------------- \n");
printf("|Arredondamento = %d  | \n",x1r);
printf("|x1b = %5.2f          | \n",x1b);
printf("|x1a = %5.2f          | \n",x1a);
printf("|iax = %d             | \n",iax);
printf("|ibx = %d             | \n",ibx);
printf("|fatry = %d           | \n",fatry);
printf("|dESi = %d            | \n",dESi);
printf("|divr = %d            | \n",divr);
printf("|fatr = %d            | \n",fatr);
printf(" ---------------------------- \n");

n=0;

//_____________________________________________________________________
//                    Definindo divisões do vetor x.
//_____________________________________________________________________
q=0;
while (n<dESf)
 {
 vetorEx[n]=q;
/*printf ("%7.2f",vetorEx[n]);
 getchar();*/
 q=q+(a1/dESf);
 n++;
 }

if(x1>=ce1)
 {
 vetor1x[0]=a1;
 n=1;
 while (n<iax*fat)
  {
  an=x1a;
  q=pow((an/a1),(1/(x1r*fat-1)));
  vetor1x[n]=x1a+a1-a1*pow(q,(n-1));
 /*printf ("%7.2f",vetor1x[n]);
  getchar(); */
  n++;
  }
```

```
      }

    q=0;
    n=0;
    while (n<divi-1)
     {
     q=q+tam/divi;
     vetor2x[n]=x1a+q;
    /*printf ("%7.2f",vetor2x[n]);
                  getchar(); */
     n++;
     }
    k=1;
    n=1;

          vetor3x[0]=(x-a1);
    while (n<(xr-ibx)*fat)
     {
     an=x-x1b;
     q=pow((an/a1),(1/((x-x1r)*fat-1)));
     vetor3x[n]=x1b-a1+a1*pow(q,(k-1));
    /*printf ("%7.2f",vetor3x[n]);
     getchar();*/
     n++;
     k++;
     }

    n=0;
    q=(x-a1);
    while (n<dESf)
     {
     q=q+(a1/dESf);
     vetorSx[n]=q;
    /*printf ("%7.2f",vetorSx[n]);
     getchar(); */
     n++;
     }
//_____________________________________________________________

//_____________________________________________________________
//                    Organização do vetor x.
//_____________________________________________________________

    for (j=0;j<iax*fat;j++)
     for (i=0;i<iax*fat-1;i++)
      {
      if (vetor1x[i]>=vetor1x[i+1])
```

```
        {
        Aux=vetor1x[i+1];
        vetor1x[i+1]=vetor1x[i];
        vetor1x[i]=Aux;
        }
       }

     for (j=0;j<divi-1;j++)
      for (i=0;i<divi-2;i++)
        {
        if (vetor2x[i]>=vetor2x[i+1])
         {
         Aux=vetor2x[i+1];
         vetor2x[i+1]=vetor2x[i];
         vetor2x[i]=Aux;
         }
        }

     for (j=0;j<(xr-ibx)*fat;j++)
      for (i=0;i<(xr-ibx)*fat-1;i++)
        {
        if (vetor3x[i]>=vetor3x[i+1])
         {
         Aux=vetor3x[i+1];
         vetor3x[i+1]=vetor3x[i];
         vetor3x[i]=Aux;
         }
        }
//_____________________________________________________________________

     op='N';

     printf("=================INFORMACAO=================== \n");
     printf("|              O obstaculo esta compreendido entre o inter-  | \n");
     printf("|          valo de:                                          | \n");
     printf("|           %7.2f <= OBSTACULO <= %7.2f                      | \n",x1a,x1b);
     printf("============================================================ \n");

//_____________________________________________________________________
//   Imprimindo na tela os resultados obtido da divisões do vetor x.
//_____________________________________________________________________
     printf("=================INFORMACAO=================== \n");
     printf("         Identificando os nos que se apresentam na ma-      \n");
     printf("       lha.                                                 \n");
     printf("============================================================ \n");

     for (i=0;i<iax*fat;i++)
```

```
      printf ("%7.2f\n",vetor1x[i]);

     for (i=0;i<divi-1;i++)
      printf ("%7.2f\n",vetor2x[i]);

     for (i=0;i<(xr-ibx)*fat;i++)
      printf ("%7.2f\n",vetor3x[i]);

     printf("=============================================================== \n");
//_____________________________________________________________________________

     }
     else
//_____________________________________________________________________________
//   Condição para que o deslocamento esteja contido no vetor x.
//_____________________________________________________________________________
      {
      if(x1<=ce1)
       {
       printf("=================INFORMACAO================================ \n");
       printf("|            O valor inserido nao possibilita o desenvol-      | \n");
       printf("       |vimento da malha, menor valor possivel para a          | \n");
       printf("|      localizacao e %5.2f.                                     | \n",ce1);
       printf("=============================================================== \n");
       }
      if(x1>=ce2)
       {
       printf("=================INFORMACAO================================ \n");
       printf("|            O valor inserido nao possibilita o desenvol-      | \n");
       printf("|      vimento da malha, maior valor possivel para a           | \n");
       printf("|      localizacao e %5.2f.                                     | \n",ce2);
       printf("=============================================================== \n");
       }
      printf("Deseja tentar novamente(S,N)?\n");
      scanf("%c",&op);
      op=(char)getchar();
      }
//_____________________________________________________________________________

    yr=(int)floor(y+.5);
    printf("=================INFORMACAO================================ \n");
    printf("|            O tamanho do obstaculo sera estabelecido co-      | \n");
    printf("|     mo sendo 8/10 do valor da altura.                         | \n");
    printf("=============================================================== \n");

    tamy=y*0.8;
    y1=y*.5;
```

```
    y1r=(int)floor(y1+.5);
    y1a=(float)y1-(y*0.8)/2;
    y1b=(float)y1+(y*0.8)/2;
    iay=(int)floor(y1a+.5);
    iby=(int)floor(y1b+.5);

    al=(int)floor(iay*faty);
    al1=(int)floor(tamy*faty);
    al2=(int)floor((y-iby)*faty);

    vetor1y = (float *) malloc ((al*sizeof(float))+1);
    vetor2y = (float *) malloc ((al1*sizeof(float))-1);
    vetor3y = (float *) malloc ((al2*sizeof(float))+1);

    printf("\n  Dados        \n");
    printf(" ------------------- \n");
    printf("|y1r = %d          | \n",y1r);
    printf("|y1a = %5.2f      | \n",y1a);
    printf("|y1b = %5.2f      | \n",y1b);
    printf("|ia = %d          | \n",iay);
    printf("|ib = %d          | \n",iby);
    printf("|al = %d          | \n",al);
    printf("|al1 = %d         | \n",al1);
    printf("|al2 = %d         | \n",al2);
    printf(" ------------------ \n");

//_____________________________________________________________________
//          Definindo as divisões do vetor y.
//_____________________________________________________________________
    n=0;
    k=0.00;

    while (n<=iay*faty)
     {
     vetor1y[n]=k;
     k=k+(y1a/(iay*faty));
     n++;
     }

    n=0;
    k=y1a;

    while (n<tamy*faty-1)
     {
     k=k+((y1b-y1a)/((iby-iay)*faty));
     vetor2y[n]=k;
     n++;
```

```
    }

   n=0;
   k=y1b;

   while (n<=(y-iby)*faty)
    {
    vetor3y[n]=k;
    k=k+((y-y1b)/((yr-iby)*faty));
    n++;
    }
//_____________________________________________________________
//_____________________________________________________________
//                        Organizando dados.
//_____________________________________________________________

   for (j=0;j<=iay;j++)
    for (i=0;i<=iay-1;i++)
     {
     if (vetor1y[i]>=vetor1y[i+1])
      {
      Aux=vetor1y[i+1];
      vetor1y[i+1]=vetor1y[i];
      vetor1y[i]=Aux;
      }
     }

   for (j=0;j<(iby-iay)-1*faty;j++)
    for (i=0;i<(iby-iay)*faty-2;i++)
     {
     if (vetor2y[i]>=vetor2y[i+1])
      {
      Aux=vetor2y[i+1];
      vetor2y[i+1]=vetor2y[i];
      vetor2y[i]=Aux;
      }
     }

   for (j=0;j<=(yr-iby)*faty;j++)
    for (i=0;i<=(yr-iby)*faty-1;i++)
     {
     if (vetor3y[i]>=vetor3y[i+1])
      {
      Aux=vetor3y[i+1];
      vetor3y[i+1]=vetor3y[i];
      vetor3y[i]=Aux;
      }
```

```
        }
//_____________________________________________________________________

    printf("=================INFORMACAO=================== \n");
    printf("|                 O obstaculo esta compreendido entre o inter-    | \n");
    printf("|          valo de:                                               | \n");
    printf("|              %7.2f <= OBSTACULO <= %7.2f                        | \n",y1a,y1b);
    printf("=================================================== \n");
//_____________________________________________________________________
//          Imprimindo na tela os resultados obtido das divisões do vetor y.
//_____________________________________________________________________

    printf("=================INFORMACAO=================== \n");
    printf("            Identificando os nos que se apresentam na ma-       \n");
    printf("         lha.                                                     \n");
    printf("=================================================== \n");

    for (i=0;i<=iay*faty;i++)
     printf ("%7.2f\n",vetor1y[i]);

    for (i=0;i<(iby-iay)*faty-1;i++)
     printf ("%7.2f\n",vetor2y[i]);

    for (i=0;i<=(y-iby)*faty;i++)
     printf ("%7.2f\n",vetor3y[i]);

    printf("=================================================== \n");
//_____________________________________________________________________
//        Executando loop no nome do arquivo para armazenamento de dados.
//_____________________________________________________________________

    strcpy(chartemp,"Coordenada_plotar_");
    strcpy(auxiliar,chartemp);

    sprintf (indice,"%d",cont);
    strcat (chartemp,indice);
    strcat (chartemp,".xls");
    printf("%s\n",chartemp);
    saida = fopen (chartemp,"w");
    strcpy(chartemp,auxiliar);
//_____________________________________________________________________

//_____________________________________________________________________
//                        Imprimindo as coordenada dos pontos.
//_____________________________________________________________________
    printf("\n=================================================== \n");
    printf("                    RESULTADOS                          \n");
```

```
printf("          Coordenada dos pontos contido na malha.          \n");
printf("═══════════════════════════════════════════════════════════ \n");

for(i=0;i<dESf;i++)//Definindo intervalo.
 for(j=0;j<(iby-iay)*faty-1;j++)//Definindo intervalo.
   {
   fprintf(saida,"%7.2f %7.2f ",vetorEx[i],vetor2y[j]);
   fprintf(saida,"\n");
   printf("(%5.2f , %5.2f) ",vetorEx[i],vetor2y[j]);
   getchar();
   }

for(i=0;i<iax*fat;i++)//Definindo intervalo.
 for(j=0;j<=iay*faty;j++)//Definindo intervalo.
   {
   fprintf(saida,"%7.2f %7.2f ",vetor1x[i],vetor1y[j]);
   fprintf(saida,"\n");
   printf("(%5.2f , %5.2f) ",vetor1x[i],vetor1y[j]);
   getchar();
   }

for(i=0;i<iax*fat;i++)//Definindo intervalo.
 for(j=0;j<(iby-iay)*faty-1;j++)//Definindo intervalo.
   {
   fprintf(saida,"%7.2f %7.2f ",vetor1x[i],vetor2y[j]);
   fprintf(saida,"\n");
   printf("(%5.2f , %5.2f) ",vetor1x[i],vetor2y[j]);
   getchar();
   }

for(i=0;i<iax*fat;i++)//Definindo intervalo.
 for(j=0;j<=(y-iby)*faty;j++)//Definindo intervalo.
   {
   fprintf(saida,"%7.2f %7.2f ",vetor1x[i],vetor3y[j]);
   fprintf(saida,"\n");
   printf("(%5.2f , %5.2f) ",vetor1x[i],vetor3y[j]);
   getchar();
   }

for(i=0;i<divi-1;i++)//Definindo intervalo.
 for(j=0;j<=iay*faty;j++)//Definindo intervalo.
   {
   fprintf(saida,"%7.2f %7.2f ",vetor2x[i],vetor1y[j]);
   fprintf(saida,"\n");
   printf("(%5.2f , %5.2f) ",vetor2x[i],vetor1y[j]);
   getchar();
   }
```

```
for(i=0;i<divi-1;i++)//Definindo intervalo.
 for(j=0;j<=(y-iby)*faty;j++)//Definindo intervalo.
   {
   fprintf(saida,"%7.2f %7.2f",vetor2x[i],vetor3y[j]);
   fprintf(saida,"\n");
   printf("(%5.2f, %5.2f) ",vetor2x[i],vetor3y[j]);
   getchar();
   }

for(i=0;i<(xr-ibx)*fat;i++)//Definindo intervalo.
 for(j=0;j<=iay*faty;j++)//Definindo intervalo.
   {
   fprintf(saida,"%7.2f %7.2f",vetor3x[i],vetor1y[j]);
   fprintf(saida,"\n");
   printf("(%5.2f, %5.2f) ",vetor3x[i],vetor1y[j]);
   getchar();
   }

for(i=0;i<(xr-ibx)*fat;i++)//Definindo intervalo.
 for(j=0;j<(iby-iay)*faty-1;j++)//Definindo intervalo.
   {
   fprintf(saida,"%7.2f %7.2f",vetor3x[i],vetor2y[j]);
   fprintf(saida,"\n");
   printf("(%5.2f, %5.2f) ",vetor3x[i],vetor2y[j]);
   getchar();
   }

for(i=0;i<(xr-ibx)*fat;i++)//Definindo intervalo.
 for(j=0;j<=(y-iby)*faty;j++)//Definindo intervalo.
   {
   fprintf(saida,"%7.2f %7.2f",vetor3x[i],vetor3y[j]);
   fprintf(saida,"\n");
   printf("(%5.2f, %5.2f) ",vetor3x[i],vetor3y[j]);
   getchar();
   }

for(i=0;i<dESf;i++)//Definindo intervalo.
 for(j=0;j<(iby-iay)*faty-1;j++)//Definindo intervalo.
   {
   fprintf(saida,"%7.2f %7.2f",vetorSx[i],vetor2y[j]);
   fprintf(saida,"\n");
   printf("(%5.2f, %5.2f) ",vetorSx[i],vetor2y[j]);
   getchar();
   }

printf("\nInteracao = %d\n",cont);
```

```
    fclose(saida);
    }

  printf("=============================================================== \n\n");
//_____________________________________________________________________________

  fclose(entr);
  }
 while(op=='S');

 printf("\nFinalizando.\n\n");
 getchar();
 return(0);
 }
```

8.5 CODIGO 3D:

```
//
//_____________________________________________________________________
//|     PROGRAMA INSTITUCIONALDE DE BOLSA DE INICIAÇÃO CIENTIFICA      |
//|                          GERADOR DE MALHA 3D.                      |
//|                                                                    |
//|              (Programa que tem por finalidade definir os pontos da malha.)  |
//|                                                                    |
//| Autor: JORGE LUIZ DEMETRIO.                                        |
//| Orientador: JERONIMO S. TRAVELHO.                                  |
//|                                                                    |
//| Data: Maio/2004.                                                   |
//|                                                                    |
//| Compilador: CC Linguagem C.                                        |
//|____________________________________________________________________|

//_____________________________________________________________________
//                  Incluindo biblioteca.
//_____________________________________________________________________
#include <stdio.h>
#include <malloc.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>
//_____________________________________________________________________

main ()
  {
//_____________________________________________________________________
//                  Declaracao de variaveis.
//_____________________________________________________________________
  char indice[40],chartemp[40],auxiliar[40],op;
  float *vetor1x,*vetor2x,*vetor3x,*vetorEx,*vetorSx;
  float *vetor1y,*vetor2y,*vetor3y,*posicao,*angulo;
  int i,j,xr,yr,iax,ibx,iay,iby,n,al,al1,al2,u,cont,m,a;
  float x1a,x1b,y1a,y1b,fat,faty,divi,dESf,k,tam,x,y,quad,b;
  int fatr,divr,dESi,x1r,fatry,y1r,aa;
  float x1,ce1,ce2,an,a1,q,Aux,y1,tamy;
  FILE *saida;
  FILE *entr;
//_____________________________________________________________________

  printf("\nInicializando...\n\n");

  printf(" _____________________________________________________ \n");
  printf("|              GERADOR DE MALHA 3D.                   |\n");
```

```
 printf("| (Programa que tem por finalidade definir os pontos da malha.)          |\n");
 printf("|                                                                        |\n");
 printf("|  Autor: JORGE LUIZ DEMETRIO.                                           |\n");
 printf("|  Orientador: JERONIMO S. TRAVELHO.                                     |\n");
 printf("|                                                                        |\n");
 printf("|  Data: Abril/2004.                                                     |\n");
 printf("|                                                                        |\n");
 printf("|  Compilador: CC Linguagem C.                                           |\n");
 printf("|________________________________________________________________________|\n");
 printf("\n");

/*printf("\nDigite o valor do comprimento do vetor x.\n");
  scanf("%f",&x);  */
//_________________________________________________________________________
//            Valor atribuido ao comprimento do vetor x.
//_________________________________________________________________________
   x=10;
//_________________________________________________________________________

  xr=(int)floor(x+.5);
/* do
   {
   printf("\nDigite o fator que multiplicara o numero pon-\n");
   printf("tos do vetor x.\n");
   scanf("%f",&fat);
   fatr=(int)floor(fat);
   if (fatr!=fat)
     {
     printf("\nO valor do fator deve ser um valor inteiro,\n");
     printf(" caso resposta for nao sera adotado 1.\n");
     printf("\nDeseja tentar novamente(S,N)?\n");
     scanf("%c",&op);
     op=(char)getchar();
     fatr=1;
     }
     else
       op='N';
   }
  while(op=='S');        */

//_________________________________________________________________________
//          Valor atribuido ao fator que multiplicara o vetor x.
//_________________________________________________________________________
  fat=5;
  fatr=5;
//_________________________________________________________________________
   /*
```

```
do
  {
  printf("\nDigite o numero de divisoes do intervalo onde\n");
  printf("esta contido o obstaculo.\n");
  scanf("%f",&divi);
  divr=(int)floor(divi);
  if (divr!=divi)
    {
    printf("\nO valor da divisoes deve ser um valor inteiro,\n");
    printf(" caso resposta for nao sera adotado 0.\n");
    printf("\nDeseja tentar novamente(S,N)?\n");
    scanf("%c",&op);
    op=(char)getchar();
    divr=0;
    }
    else
      op='N';
  }
 while (op=='S'); */

//_____________________________________________________________________
//      Numero atribuido a quantidade de divisoes no intervalo
//              em que o obstaculo esta contido.
//_____________________________________________________________________
 divi=20;
 divr=20;
//_____________________________________________________________________
 /*
 do
  {
  printf("\nDigite o numero  de  divisoes do intervalo da\n");
  printf("entrada e saida.\n");
  scanf("%f",&dESf);
  dESi=(int)floor(dESf);
  if (dESi!=dESf)
    {
    printf("\nO valor da divisoes deve ser um valor inteiro,\n");
    printf(" caso resposta for nao sera adotado 1.\n");
    printf("\nDeseja tentar novamente(S,N)?\n");
    scanf("%c",&op);
    op=(char)getchar();
    dESi=1;
    }
    else
      op='N';
  }
 while (op=='S');  */
```

```
//_____________________________________________________________________
//      Numero atribuido quantidade de divisoes na entrada e saida.
//_____________________________________________________________________
 dESi=4;
 dESf=4;
//_____________________________________________________________________
  /*
 printf("\nDeterminar o tamanho do corte no vetor x.\n");
 scanf("%f",&a1);      */

//_____________________________________________________________________
//      Numero atribuido ao corte da valvula.
//_____________________________________________________________________
 a1=1;
//_____________________________________________________________________
   /*
 printf("\nDigite o valor do comprimento do vetor y.\n");
 scanf("%f",&y);  */
//_____________________________________________________________________
//        Valor atribuido a altura do vetor y.
//_____________________________________________________________________
   y=10;
//_____________________________________________________________________
     /*
 do
   {
  printf("\nDigite o fator que multiplicara o numero pon-\n");
  printf("tos do vetor y.\n");
  scanf("%f",&faty);
  faty=3;
  fatry=floor(faty);
  if (fatry!=faty)
    {
    printf("\nO valor do fator deve ser um valor inteiro,\n");
    printf(" caso resposta for nao sera adotado 1.\n");
    printf("\nDeseja tentar novamente(S,N)?\n");
    scanf("%c",&op);
    op=(char)getchar();
    fatry=1;
    }
    else
      op='N';
  }
 while(op=='S');    */
//_____________________________________________________________________
//        Valor atribuido ao fator que multiplicara o vetor y.
```

```
//_____________________________________________________________________
    faty=5;
    fatry=5;
//_____________________________________________________________________
  /*
 printf("\nDigite o  numero corte do vetor z.\n");
 scanf("%d",&aa);
 */
 aa=24;
 /*
 printf("\nDigite o tamanho da aresta do cubo.\n");
 scanf("%f",&quad);
 */
 quad=1;

 printf("================INFORMACAO=========================== \n");
 printf("|          O tamanho do obstaculo sera estabelecido co-          | \n");
 printf("|mo sendo 1/10 do valor comprimento.                              | \n");
 printf("=====================================================  \n");
 tam=x*0.1;

 do
  {
  entr=fopen("Dado.txt","r");
  fscanf(entr,"%d",&u);
  posicao = (float*) malloc (u*sizeof (float) );
  for (j=0;j<u;j++)
   fscanf (entr,"%f",&posicao[j]);

  for (i=0;i<u;i++)
   printf ("%7.2f\n",posicao[i]);

  x1=0;
  cont=0;
  while(x1!=posicao[u-1])
   {
   x1=posicao[cont];
   cont++;
   ce1=(tam/2)+a1;
   ce2=(x-(tam/2)-a1);
   x1r=(int)floor(x1+.5);

   if (x1>=ce1 && x1<=ce2)
    {
    x1a=(float)x1-(x*0.1)/2;
    iax=(int)floor(x1a+.5);
    x1b=(float)x1+(x*0.1)/2;
```

```
ibx=(int)floor(x1b-.5);

if(x1r>x1)
 {
 iax=iax+1;
 ibx=ibx+1;
 }

vetor1x = (float *) malloc (iax*fat*sizeof (float));
vetor2x = (float *) malloc ((divi*sizeof (float))-1);
vetor3x = (float *) malloc ((xr-ibx)*fat*sizeof (float));
vetorEx = (float *) malloc (dESf*sizeof (float));
vetorSx = (float *) malloc (dESf*sizeof (float));

printf("\n  Dados          \n");
printf(" ---------------------  \n");
printf("|Arredondamento = %d | \n",x1r);
printf("|x1b = %5.2f       | \n",x1b);
printf("|x1a = %5.2f       | \n",x1a);
printf("|iax = %d          | \n",iax);
printf("|ibx = %d          | \n",ibx);
printf(" ---------------------  \n");

n=0;

//_____________________________________________________________________
//              Definindo divisões do vetor x.
//_____________________________________________________________________
q=0;
while (n<dESf)
 {
 vetorEx[n]=q;
/*printf ("%7.2f",vetorEx[n]);
 getchar();*/
 q=q+(a1/dESf);
 n++;
 }

if(x1>=ce1)
 {
 vetor1x[0]=a1;
 n=1;
 while (n<iax*fat)
  {
  an=x1a;
  q=pow((an/a1),(1/(x1r*fat-1)));
  vetor1x[n]=x1a+a1-a1*pow(q,(n-1));
```

```
 /*printf ("%7.2f",vetor1x[n]);
   getchar(); */
   n++;
   }
  }

 q=0;
 n=0;
 while (n<divi-1)
  {
  q=q+tam/divi;
  vetor2x[n]=x1a+q;
 /*printf ("%7.2f",vetor2x[n]);
                getchar(); */
  n++;
  }
 k=1;
 n=1;

       vetor3x[0]=(x-a1);
 while (n<(xr-ibx)*fat)
  {
  an=x-x1b;
  q=pow((an/a1),(1/((x-x1r)*fat-1)));
  vetor3x[n]=x1b-a1+a1*pow(q,(k-1));
 /*printf ("%7.2f",vetor3x[n]);
  getchar();*/
  n++;
  k++;
  }

 n=0;
 q=(x-a1);
 while (n<dESf)
  {
  q=q+(a1/dESf);
  vetorSx[n]=q;
 /*printf ("%7.2f",vetorSx[n]);
  getchar(); */
  n++;
  }
//_____________________________________________________________

//_____________________________________________________________
//                    Organaização do vetor x.
//_____________________________________________________________
```

```
      for (j=0;j<iax*fat;j++)
       for (i=0;i<iax*fat-1;i++)
         {
         if (vetor1x[i]>=vetor1x[i+1])
           {
           Aux=vetor1x[i+1];
           vetor1x[i+1]=vetor1x[i];
           vetor1x[i]=Aux;
           }
         }

      for (j=0;j<divi-1;j++)
       for (i=0;i<divi-2;i++)
         {
         if (vetor2x[i]>=vetor2x[i+1])
           {
           Aux=vetor2x[i+1];
           vetor2x[i+1]=vetor2x[i];
           vetor2x[i]=Aux;
           }
         }

      for (j=0;j<(xr-ibx)*fat;j++)
       for (i=0;i<(xr-ibx)*fat-1;i++)
         {
         if (vetor3x[i]>=vetor3x[i+1])
           {
           Aux=vetor3x[i+1];
           vetor3x[i+1]=vetor3x[i];
           vetor3x[i]=Aux;
           }
         }
//_____________________________________________________________________

      op='N';

      printf("================INFORMACAO=========================== \n");
      printf("|          O obstaculo esta compreendido entre o inter-          | \n");
      printf("|valo de:                                                        | \n");
      printf("|    %7.2f <= OBSTACULO <=  %7.2f                              | \n",x1a,x1b);
      printf("================================================================\n");


//_____________________________________________________________________
//    Imprimindo na tela os resultados obtido das divisões do vetor x.
//_____________________________________________________________________
      printf("================INFORMACAO=========================== \n");
```

```
    printf("              Identificando os nos que se apresentam na ma-          \n");
    printf("lha.                                                                  \n");
    printf("=======================================================================\n");
    for (i=0;i<iax*fat;i++)
     printf ("%7.2f\n",vetor1x[i]);

    for (i=0;i<divi-1;i++)
     printf ("%7.2f\n",vetor2x[i]);

    for (i=0;i<(xr-ibx)*fat;i++)
     printf ("%7.2f\n",vetor3x[i]);

    printf("=======================================================================\n");
//_____________________________________________________________________________

    }
    else
//_____________________________________________________________________________
//    Condicao para que o deslocamento esteja contido no vetor x.
//_____________________________________________________________________________
     {
     if(x1<=ce1)
      {
      printf("=================INFORMACAO============================== \n");
      printf("|            O valor inserido nao possibilita o desenvol -      | \n");
      printf("|vimento da malha, menor valor possivel para a                  | \n");
      printf("|localizacao e %5.2f.                                         | \n",ce1);
      printf("\n======================================================== \n");
      }
     if(x1>=ce2)
      {
      printf("=================INFORMACAO============================== \n");
      printf("|              O valor inserido nao possibilita o desenvol-     | \n");
      printf("|vimento da malha, maior valor possivel para a                  | \n");
      printf("|localizacao e %5.2f.                                         | \n",ce2);
      printf("======================================================== \n");
      }
     printf("Deseja tentar novamente(S,N)?\n");
     scanf("%c",&op);
     op=(char)getchar();
     }
//_____________________________________________________________________________

    yr=(int)floor(y+.5);
    printf("=================INFORMACAO============================== \n");
    printf("|          O tamanho do obstaculo sera estabelecido co-         | \n");
    printf("|mo sendo 8/10 do valor ad altura.                              | \n");
```

```
printf("=========================================================== \n");
tamy=y*0.8;
y1=y*.5;
y1r=(int)floor(y1+.5);
y1a=(float)y1-(y*0.8)/2;
y1b=(float)y1+(y*0.8)/2;
iay=(int)floor(y1a+.5);
iby=(int)floor(y1b+.5);

al=(int)floor((iay/2)*faty);
al1=(int)floor((tamy/2)*faty);
al2=(int)floor((iay/2)*faty);

vetor1y = (float *) malloc ((al*sizeof(float))+1);
vetor2y = (float *) malloc ((al1*sizeof(float))-1);
vetor3y = (float *) malloc ((al2*sizeof(float))+1);

printf("\n  Dados        \n");
printf(" ------------------- \n");
printf("|y1r = %d         | \n",y1r);
printf("|y1a = %5.2f      | \n",y1a);
printf("|y1b = %5.2f      | \n",y1b);
printf("|ia = %d          | \n",iay);
printf("|ib = %d          | \n",iby);
printf("|al = %d          | \n",al);
printf("|al1 = %d         | \n",al1);
printf("|al2 = %d         | \n",al2);
printf(" ------------------- \n");

//_____________________________________________________________________
//          Definindo as divisões do vetor y.
//_____________________________________________________________________
   n=0;
   k=0.00;

   while (n<=(iby-y1)*faty)
    {
    vetor1y[n]=k;
    k=k+((y1b-y1)/((iby-y1)*faty));
    printf ("%7.2f",vetor1y[n]);
    getchar();
    n++;
    }

   b=n;
   n=0;
```

```
while (n<(y-y1b)*faty)
 {
 vetor2y[n]=k;
 k=k+((y-y1b)/((y-iby)*faty));
 printf ("%7.2f",vetor2y[n]);
 getchar();
 n++;
 }

n=0;

while (n<=(iby-y1)*faty)
 {
 vetor3y[n]=vetor1y[n];
 printf ("%7.2f",vetor3y[n]);
 getchar();
 n++;
 }


//_____________________________________________________________________

 printf("================INFORMACAO=================== \n");
 printf("|        O obstaculo esta compreendido entre o inter-          | \n");
 printf("|valo de:                                                      | \n");
 printf("|   %7.2f <= OBSTACULO <= %7.2f                              | \n",y1a,y1b);
 printf("================================================ \n");
//_____________________________________________________________________
// Imprimindo na tela os resultados obtido das divisões do vetor y.
//_____________________________________________________________________

 printf("================INFORMACAO=================== \n");
 printf("Identificando os nos que se apresentam na ma-                \n");
 printf("lha.                                                         \n");
 printf("================================================ \n");

 for (i=0;i<=(iby-y1)*faty;i++)
  printf ("%7.2f\n",vetor1y[i]);

 printf ("\n");

 for (i=0;i<(y-y1b)*faty;i++)
  printf ("%7.2f\n",vetor2y[i]);

 printf ("\n");

 for (i=0;i<=(iby-y1)*faty;i++)
```

```
    printf ("%7.2f\n",vetor3y[i]);

   printf("=============================================================== \n");
//_____________________________________________________________________________


//_____________________________________________________________________________
//   Executando loop no nome do arquivo para armazenamento de dados.
//_____________________________________________________________________________

    strcpy(chartemp,"Coordenada_plotar_");
    strcpy(auxiliar,chartemp);

    sprintf (indice,"%d",cont);
    strcat (chartemp,indice);
    strcat (chartemp,".xls");
    printf("%s\n",chartemp);
    saida = fopen (chartemp,"w");
    strcpy(chartemp,auxiliar);
//_____________________________________________________________________________

//_____________________________________________________________________________
//       Imprimindo as coordenada dos pontos.
//_____________________________________________________________________________

    printf("\n=============================================================== \n");
    printf("                          RESULTADOS                             \n");
    printf("       Coordenada dos pontos contido na malha.                   \n");
    printf("=============================================================== \n");

    angulo = (float*) malloc (aa*sizeof (float) );
    k=0;
    for(m=0;m<=aa;m++)
     {
     angulo[m]=k;
     k=k+(360/aa);
     printf("%f \n",angulo[m]);
     }

    for(i=0;i<dESf;i++)
     for(j=0;j<=b;j++)
      {
      if(vetor1y[j]>(quad/2))
       {
       for(m=0;m<aa;m++)
        {
```

```
          fprintf(saida,"%7.2f %7.2f %7.2f
",vetor1y[j]*sin(angulo[m]),vetor1y[j]*cos(angulo[m]),vetorEx[i]);
          fprintf(saida,"\n");
          fprintf(saida,"%7.2f %7.2f %7.2f",-
vetor1y[j]*sin(angulo[m]),vetor1y[j]*cos(angulo[m]),vetorEx[i]);
          fprintf(saida,"\n");
          fprintf(saida,"%7.2f %7.2f %7.2f",vetor1y[j]*sin(angulo[m]),-
vetor1y[j]*cos(angulo[m]),vetorEx[i]);
          fprintf(saida,"\n");
          fprintf(saida,"%7.2f %7.2f %7.2f",-vetor1y[j]*sin(angulo[m]),-
vetor1y[j]*cos(angulo[m]),vetorEx[i]);
          fprintf(saida,"\n");
          printf("\n etapa 3 \n");
          }
         }
        if(vetor2y[j]<=(quad/2))
         {
         for(a=0;a<=b;a++)
          {
          if(vetor3y[a]<=(quad/2))
           {
           fprintf(saida,"%7.2f %7.2f %7.2f",vetor3y[a],vetor2y[j],vetorEx[i]);
           fprintf(saida,"\n");
           fprintf(saida,"%7.2f %7.2f %7.2f",-vetor3y[a],vetor2y[j],vetorEx[i]);
           fprintf(saida,"\n");
           fprintf(saida,"%7.2f %7.2f %7.2f",vetor3y[a],-vetor2y[j],vetorEx[i]);
           fprintf(saida,"\n");
           fprintf(saida,"%7.2f %7.2f %7.2f",-vetor3y[a],-vetor2y[j],vetorEx[i]);
           fprintf(saida,"\n");
           printf("\n etapa \n");
           }
          printf("\n etapa 2 \n");
          }
         }
        }

     fprintf(saida,"\n fim teste.\n");



     printf("\nInteracao = %d\n",cont);
     printf("========================================================== \n\n");
     fclose(saida);
  //_____________________________________________________________________

     }
```

```
 fclose(entr);
 }
while(op=='S');

printf("\nFinalizando.\n\n");
getchar();
return(0);
}
```

8.6 CODIGO MODELO FISICO:

```
//_____________________________________________________________________
//|        Programa Institucional Bolsa de Iniciacao Cientifica - CNPQ/PIBIC.  |
//|                        (Modelo matematico e interpolacao.)                 |
//| Autor: Jorge Luiz Demetrio.                                                |
//| Orientador: Jeronimo S. Travelho.                                          |
//| Data: marco/2004.                                                          |
//| Compilador cc Linguagem C.                                                 |
//|____________________________________________________________________________|

//_____________________________________________________________________
//                  Incluindo biblioteca.
//_____________________________________________________________________
#include<stdio.h>
#include<math.h>
#include<malloc.h>
#include<stdlib.h>
//_____________________________________________________________________

main()
  {
  FILE *dado,*dado_coordenada,*dado_veloc,*dado_aceler;
  float *deslocamento,*velocidade,*aceleracao,*vetorx,*vetory,*veloc_aux;
  float *aceler_aux;
  float forca,massa,tempo,delta_t,c0,c1;
  int i,j,n,alocador,linha;

  printf("\n Inicializando...\n\n");
  printf("_____________________________________________________________\n");
  printf("|                        MODELADOR.                           |\n");
  printf("|                (Modelo matematico e interpolacao.)          |\n");
  printf("| Autor: Jorge Luiz Demetrio.                                 |\n");
  printf("| Orientador: Jeronimo S. Travelho.                           |\n");
  printf("| Data: marco/2004.                                           |\n");
  printf("| Compilador cc Linguagem C.                                  |\n");
  printf("|_____________________________________________________________|\n");
  printf("\n");
  printf("Forneca a forca que atua no sistema.\n");
  scanf("%f",&forca);
  printf("\nForneca a massa do obstaculo.\n");
  scanf("%f",&massa);
  printf("\nForneca a variacao no tempo.\n");
  scanf("%f",&delta_t);

  alocador = floor(10/delta_t);
```

```
deslocamento = (float *) malloc (alocador*sizeof (float));
velocidade = (float *) malloc (alocador*sizeof (float));
aceleracao = (float *) malloc (alocador*sizeof (float));

n=0;
tempo=0.00;
velocidade[0]=0.00;
deslocamento[0]=0.00;

//_____________________________________________________________________
//          Criando arquivo de referencia para interpolação.
//_____________________________________________________________________

 dado=fopen("Tabela_dados.xls","w");
 do
   {
   getchar();
   velocidade[n+1]=velocidade[n]+((forca/massa)*delta_t);
   aceleracao[n]=(velocidade[n+1]-velocidade[n])/delta_t;
   deslocamento[n+1]=deslocamento[n]+(velocidade[n+1]*delta_t);
   fprintf(dado,"%5.2f %5.2f %5.2f %d
\n",deslocamento[n],velocidade[n],aceleracao[n],n);
   printf("velocidade = %5.2f\n",velocidade[n+1]);
   printf("aceleracao = %5.2f\n",aceleracao[n]);
   printf("deslocamento = %5.2f\n",deslocamento[n+1]);
   tempo=tempo+delta_t;
   n++;
   }
  while(deslocamento[n]<=10+1);

  alocador=n;
  fclose(dado);
 //_____________________________________________________________________

 //_____________________________________________________________________
 //       Fazendo leitura do arquivo para interpolação.
 //_____________________________________________________________________

   dado_coordenada=fopen("Coordenada_plotar.xls","r");
   fscanf(dado_coordenada,"%d\n",&linha);
   printf("%d \n",linha);

   vetorx = (float *) malloc (linha*sizeof (float));
   vetory = (float *) malloc (linha*sizeof (float));
   veloc_aux = (float *) malloc (linha*sizeof (float));
   aceler_aux = (float *) malloc (linha*sizeof (float));
```

```
for(i=0;i<linha;i++)//Definindo intervalo.
  {
  fscanf(dado_coordenada ,"%f %f\n",&vetorx[i],&vetory[i]);
  printf("%f %f\n",vetorx[i],vetory[i]);
  }

 fclose(dado_coordenada );
//_____________________________________________________________

//_____________________________________________________________
//            Procedimento para interpolacao velocidade.
//_____________________________________________________________

 for (i=0;i<linha;i++)
  for (j=0;j<alocador-1;j++)
    {
    if(vetorx[i]>=deslocamento[j]||vetorx[i]<=deslocamento[j+1])
      {
      c0=velocidade[j];
      c1=((velocidade[j+1]-c0)/(deslocamento[j+1]-deslocamento[j]));
      veloc_aux[i]=c0+(c1*vetorx[i])-(c1*deslocamento[j]);
      }
    }
//_____________________________________________________________

//_____________________________________________________________
//      Criando arquivo com dados de interpolacao da velocidade.
//_____________________________________________________________

  dado_veloc = fopen("Dados_x_y_v.xls","w");

  for(i=0;i<linha;i++)
    {
    fprintf(dado_veloc,"%5.2f %5.2f %5.2f\n",vetorx[i],vetory[i],veloc_aux[i]);
    printf("%5.2f %5.2f %5.2f \n",vetorx[i],vetory[i],veloc_aux[i]);
    getchar();
    }

  fclose(dado_veloc);
//_____________________________________________________________

//_____________________________________________________________
//            Procedimento para interpolacao da aceleracao.
//_____________________________________________________________

  for (i=0;i<linha;i++)
```

```
  for (j=0;j<alocador-1;j++)
    {
    if(vetorx[i]>=deslocamento[j]||vetorx[i]<=deslocamento[j+1])
      {
      c0=aceleracao[j];
      c1=((aceleracao[j+1]-c0)/(deslocamento[j+1]-deslocamento[j]));
      aceler_aux[i]=c0+(c1*vetorx[i])-(c1*deslocamento[j]);
      }
    }
//_____________________________________________________________

//_____________________________________________________________
//     Criando arquivo com dados de interpolacao da aceleracao.
//_____________________________________________________________

  dado_aceler = fopen("Dados_x_y_a.xls","w");

  for(i=0;i<linha;i++)
    {
    fprintf(dado_aceler,"%5.2f %5.2f %5.2f\n",vetorx[i],vetory[i],veloc_aux[i]);
    printf("%5.2f %5.2f %5.2f \n",vetorx[i],vetory[i],aceler_aux[i]);
    getchar();
    }

  fclose(dado_aceler);
//_____________________________________________________________

  printf("\n Finalizando.\n\n");
  return (0);

  }
```

9.1 Comentários e conclusões:

O objetivo principal do trabalho foi criar um gerador de malha computacional para os casos unidimensional, bidimensional e tridimensional a partir de determinadas condições de contorno de uma válvula de retenção. A importância deste estudo está voltada principalmente no projeto e construção de sistemas para avaliar os efeitos ocasionados na válvula durante sua utilização.

A presença de camadas de turbulência no interior da válvula pode ocasionar o rompimento da tubulação fator que representa um grande perigo para as pessoas, pois pode produzir vários tipos de acidentes como já foram registrados como sendo causados por este fenômeno. Dar se a então a necessidade de se fazer uma analise das condições que incidem sobre a válvula.

O modelo físico poderá ser aplicado a esta malha para obter os resultados. Porém, não é este o objetivo deste trabalho que não visa em estudar o problema físico e iniciar o desenvolvimento de um programa de simulação de formação dos vórtice, mas sim em definir a geometria a ser adotada.

A geometria do obstáculo mantém as características que permitem a analise do escoamento turbulento em torno do corpo. O fato que torna o programa do caso unidimensional e bidimensional com qualidade é que qualitativamente os resultados obtidos com varias dimensões adotadas, tendo resultados satisfatórios e estão coerentes com a física do problema. Ainda, a

influência de cada parâmetro na forma da válvula foi relatada de modo a ter-se um maior entendimento do procedimento para sua divisão e seus efeitos sobre a forma do vórtice.

Para o caso tridimensional não foi possível gerar os resultados devido a algum tipo de erro não identificado por min na programação do mesmo.

Foi visto que os fatores que mais influenciam nas formas são: a velocidade, temperatura, a viscosidade dinâmica, densidade, rugosidade, são fatores que afetam mais significantemente na formação dos vórtices.

10.1 COMENTARIOS DE POSSÍVEIS TRABALHOS FUTUROS:

O modelo matemático é uma das partes mais importantes para a simulação, porém esta quando executada não pode ser considerada como a solução do problema. Pois se admitiríamos as hipóteses consequentemente de os resultados obtidos da simulação modelada pelas equações seria igual ao o fenômeno real, e isto não seria verdade, pois o fenômeno modelado pelas equações possui diferenças entre o fenômeno real.

Todas as etapas do fenômeno do modelo físico serão alvo para um possível trabalho posterior a este, e será descrito, de modo a abordar o desenvolvimento das equações e as hipóteses feitas em cada caso. Detalhes desde a discretização da superfície até o estabelecimento do modelo. Porém, a complexidade do fenômeno físico em si torna o modelo matemático também complexo. Acrescenta-se a isso o fato de estar-se resolvendo um escoamento em torno de um perfil que esta configurada de forma dinâmica e ter que definir as trajetórias de partículas do fluido.

10.2 Cálculo do escoamento:

Esta etapa consiste na solução de equações diferenciais que fornecem o campo de velocidades e pressões sobre a geometria. Bastante comum em problemas de hidrodinâmica, a solução do escoamento por meio de método possível para a solução.

10.3 Cálculo da camada limite:

Os métodos integrais para cálculo de camada limite hidrodinâmica consistem basicamente em propor um perfil de velocidade em função da espessura da camada e de constantes a serem determinadas. Aplicando as condições de contorno, determina-se este valor. O método sugerido para solução velocidade deste trabalho é o de Von Kármán que pode ser aplicado para a camada limite bidimensional. Este é utilizado e se mostra bastante eficaz. Uma descrição do método é feita em seguida. Considera-se a equação da quantidade de movimento para camada limite laminar.

1. MUNSON, B. R.; YOUNG, D. F.; OKIISHI, T. H., "Fundamentos da mecânica dos fluidos", ed. 2/1997.

2. CATTANI, M. S. D., "Elementos de mecânica dos fluidos", ed. 1990.

3. SAMPAIO, M. C.; SAUVÉ, J. P.; MOURA, J. A. B., "UNIX guia do usuário", ed. 1998.

4. STEINBRUCH, A.; WINTERLE, P., "Geometria analítica", ed. 2/1987.

5. VICTORINE; VIVIANE; MIZRAHI, "Treinamento em linguagem C", ed. 1/1998.

6. Professor: Marcos Aurélio Pchek Laureano, "Curso básico de programação em linguagem C". http://www.ppgia.pucpr.br/~laureano/ensino/cursos.

7. Organizado por Eder Fantini Junqueira, "Portal da linguagem C". http://www.portalc.nip.net.