This is to certify that the
thesis entitled

ELLIPTIC GRID GENERATION, SMOOTHTING, AND REFINEMENT FOR STRUCTURED AND UNSTRUCTURED MESHES

presented by

DEEPAK TIWARI

has been accepted towards fulfillment
of the requirements for the

Master of Science degree in Mechanical Engineering

Major Professor's Signature

April 21, 2004

Date

ELLIPTIC GRID GENERATION, SMOOTHING, AND REFINEMENT FOR STRUCTURED AND UNSTRUCTURED MESHES

By

Deepak Tiwari

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Department of Mechanical Engineering

2004

ABSTRACT

ELLIPTIC GRID GENERATION, SMOOTHING, AND REFINEMENT FOR STRUCTURED AND UNSTRUCTURED MESHES

By

Deepak Tiwari

Finite difference (FD), finite volume (FV), and finite-element (FE) methods are very powerful techniques for obtaining solutions to partial differential equations that govern fluid flow problems. However, in order to use these methods, it is necessary to replace the spatial domain of the problem being studied by a finite number of grid points or cells. Depending upon how the grid points or cells are connected to each other, the resulting grid is referred to as structured or unstructured. In this study, an algorithm based on elliptic partial differential equations, a technique used to generate and smooth structured grids, has been generalized to be used with both structured and unstructured grids. The algorithm implemented enables uniform local or global smoothing across all block boundaries. It also enables directional smoothing of the grid to allow for high-aspect ratio grids that are aligned with the flow direction. The algorithm developed also enables grid refinement and coarsening. Refinement and coarsening are achieved by simply inserting and deleting grid points, and redistributing them by elliptic or directional smoothing. The usefulness of the method developed is illustrated by applying it to generate, smooth, and refine grids for several example problems, including the grids for the combustion chamber of an internal combustion engine with complicated shape piston bowls and cylinder heads.

To my parents Mr. Ishwar Lal Tiwari and Mrs. Vimla Tiwari

And my sisters Mrs. Neeta Tiwari and Ms. Nilima Tiwari

for their unfailing love and support

ACKNOWLEDGMENTS

I would like to record my deep sense of gratitude and thanks to my thesis adviser and mentor Professor Tom Shih for his unfailing and continuous support in my research work. The success of my work is solely due to his able guidance and constant encouragement.

I would like to thank Dr. Allen Han and Dr. James Yi of Ford Motor Company, Scientific Research Laboratory at Dearborn, Michigan. I gained immense understanding of the aspects of an Internal Combustion Engine Design and exposure to the facilities at Ford SRL gave me an appreciation of the development efforts in the industry. I would also like to thank my friends and office-mates Mr. Robert Draper, Mr. Sri Harsha Chunduru, and Mr. Praveen Vasam for their constant encouragement and interest in my work. I would like to extend sincere thanks to Professor Farhad Jaberi and Professor Harold Schock for their suggestions and agreeing to judge my thesis work.

I would like to thank my dear friends Paramita, Asavari, Jaya, Subathra, and Tara for their support. I would finally like to thank my family for their support. I would especially like to thank my father Mr. Ishwar Lal Tiwari, mother Mrs. Vimla Tiwari, and sisters Neeta and Nilima Tiwari for their love and support.

Deepak Tiwari
Department of Mechanical Engineering
Michigan State University
East Lansing MI – 48824

Date: 02 May, 2004

TABLE OF CONTENTS

TABLE OF FIGURES

CHAPTER 1 INTRODUCTION

1.1 Preliminary

Finite difference (FD), finite volume (FV), and finite-element (FE) methods are powerful techniques for obtaining solutions to partial differential equations that govern fluid flow problems. However, in order to use these methods, it is necessary to replace the spatial domain of the problem by a finite number of grid points or cells. The process of replacing a spatial domain by a system of grid points or cells is referred to as grid generation. Grid generation is a very important part of FD, FV, and FE methods because the system of grid points or cells used strongly affects the accuracy, efficiency and ease with which these methods generate solutions. In some instances, the ability or inability to generate an "acceptable" grid system determines whether FD, FV, or FE methods can or cannot be used. An "acceptable" grid is one that would resolve not only the geometry but also the flow physics to the desired accuracy with minimal grid-induced errors. We focus this study on how to generate "acceptable" grid to facilitate FD, FV, and FE methods.

All grid systems can be classified as structured, unstructured, or mixed, depending upon how the grid points are connected to each other to form cells (see for example Refs. 1 and 2). Structured grids can be generated by algebraic or partial differential equations. Algebraic methods include the two-, four-, and six-boundary methods (Refs. 2 and 3) and multi-surface methods (Ref. 2 and 4). Partial differential equation (PDE) methods include those based on elliptic, parabolic, and hyperbolic PDEs (Ref. 4). Of the PDE methods, those based on elliptic PDEs are the most widely used because they are the most versatile. Two types of elliptic PDEs have been used, Laplace and Poisson. The Laplace equation ensures a smooth grid (i.e., it distributes grids points uniformly

throughout the domain, forming cells with nearly the same area in two dimensions or volume in three dimensions throughout the domain). In addition to smoothing, the Laplace equation guarantees no overlap in grid lines. The Poisson equation allows grid points to be clustered about any point or lines in the spatial domain. But, there is no guarantee that grid lines will not overlap. Although there is no guarantee in all cases, there are ways to define the coefficients of the Poisson equation to ensure no overlap in certain cases. It is worth mentioning here that in structured meshes these operators can be applied directionally, that is, the smoothing or clustering operations can be done selectively in a chosen vectorial functional direction.

Unstructured grids are generated by methods that are quite different from those used to generate structured grids. Unstructured grids can be classified as isotropic and anisotropic mesh (Ref. 5). The generation of anisotropic mesh is still in a state of development (see, e.g., Ref. 6). The generation of isotropic mesh is well developed with a variety of techniques (e.g., advancing front and advancing layers) and has been applied to numerous applications. All methods use the concept of Delaunay triangulation (or tetrahedralization) when forming "isotropic" cells. Delaunay triangulation (or tetrahedralization) is an effective criterion for generating "acceptable" grid. "Acceptable" cells are constructed iteratively by edge or face swapping. In 2-D, a unique isotropic mesh can be generated. In 3-D, Delaunay triangulation cannot guarantee of an "acceptable" mesh. "Slivers" (i.e., 3-D cells that appear as if they are 2-D) invariably form, and they must however, be smoothened. The use of unstructured grids with FV methods is gaining wider acceptance in industry and academia, and considerable research is still going on in this area (Refs. 4, 5, 6).

In this study it is proposed to generalize the PDE approaches developed for structured meshes to smooth, cluster, and refine unstructured meshes. In many cases, a final "acceptable" unstructured mesh is generated by repeated and alternate smoothing (PDE approach) and swapping (Delaunay) operations. Clever use of smoothing would reduce the number of iterations for the more expensive Delaunay triangulation (or tetrahedralization).

The algorithm was applied to a set of multiblock structured grid from KIVA-3V and simple two-dimensional unstructured grids. An internal combustion engine geometry was chosen to generate the initial multiblock structured mesh. The initial mesh had bad quality regions due to which KIVA would reject the input in critical cases. Such cases were typical when piston head was close to the top dead center and mesh was particularly bad with high volume ratios and skewness (mathematically defined in chapter 4). A sample two-dimensional unstructured mesh was generated to apply the generalized algorithm in unstructured meshes. The final mesh was a definite and significant improvement over the initial mesh. The final mesh was "acceptable" in terms of quantitative grid quality measure and KIVA accepted the mesh in all steps.

1.2 Objective

The objective of this study is to generalize elliptic PDE methods developed for generating structured grids for unstructured grids. More specifically, the objectives are as follows:

1. Develop an algorithm based on the Laplace equation using unstructured data format to smooth structured multi-block grids.

2. Generalize the Laplace equation to enable directional smoothing of structured grids.
3. Specify the Poisson operator to generate structured grids with clustering without overlap.
4. Generalize the Laplace equation for generating unstructured isotropic mesh.
5. Generalize the Laplace operator to enable refining and coarsening of unstructured grids based on weighted averaging.

CHAPTER 2 ALGORITHM

2.1 Introduction

In this chapter, we will describe and formulate the algorithm to smooth and cluster a pre-existing grid that guarantees no overlap by using elliptic PDEs. The most common examples of elliptic PDEs are Laplace, Poisson, and Helmholtz equations. We consider a grid system, where location of the boundary nodes are either fixed or are allowed to move in a well-defined manner. This means that we know the boundary value of the function involved in these equations, at all times. Thus the problem is defined.

We use Laplace operator to smooth the grid. We use Poisson operator to cluster the grid in a specified direction or in a region of interest in a way that it ensures smooth clustering and no overlap. We will first formulate the Laplace operator for structured and multi-block structured grid and then generalize the algorithm for unstructured grid. We then describe a generalized Poisson operator that can be applied to structured and unstructured grids.

2.2 The Laplace Operator for Multi-Block Structured Mesh

The numerical method discussed here is based on discrete laplacian. A discrete laplacian does not guarantee that there would be no overlap of grid lines in all cases. When the domain is convex (or almost convex), discrete laplacian strategy will generally produce good grids. However, for nonconvex regions and other special situations, it may produce grids that are not valid (Ref 5). Winslow (Ref 7) suggests smoothing based on weighted averaging to solve this problem. We use the concept of weighted averaging based on nodal distance to smooth the grid thus eliminating possibility of grid overlap in any case. We transform the physical domain in the Cartesian coordinate system to generalized co-

ordinate system, creating the computational domain. It is customary approach to transform the domain to generalized co-ordinate system to simplify the problem domain, but is not a requirement. The idea is to map whole domain to a square (or cube for 3D). For example in the domain each quadrilateral cell can be mapped onto a square cell in the computational domain. Similarly each hexahedral cell can be mapped onto a cubical cell in the computational domain. Use of computational domain simplifies the boundary conditions, measurements and application of constraints.

2.2.1 Elliptic Grid Generation Based on Weighted Laplacian

The position of any grid point in Cartesian coordinate system X,Y,Z can be expressed as a function of generalized coordinates ξ,η,ζ and time. Ignoring the time dependency, the harmonic mapping can be represented as:

$$(X,Y,Z) \Leftrightarrow (\xi,\eta,\zeta) \tag{2.1}$$

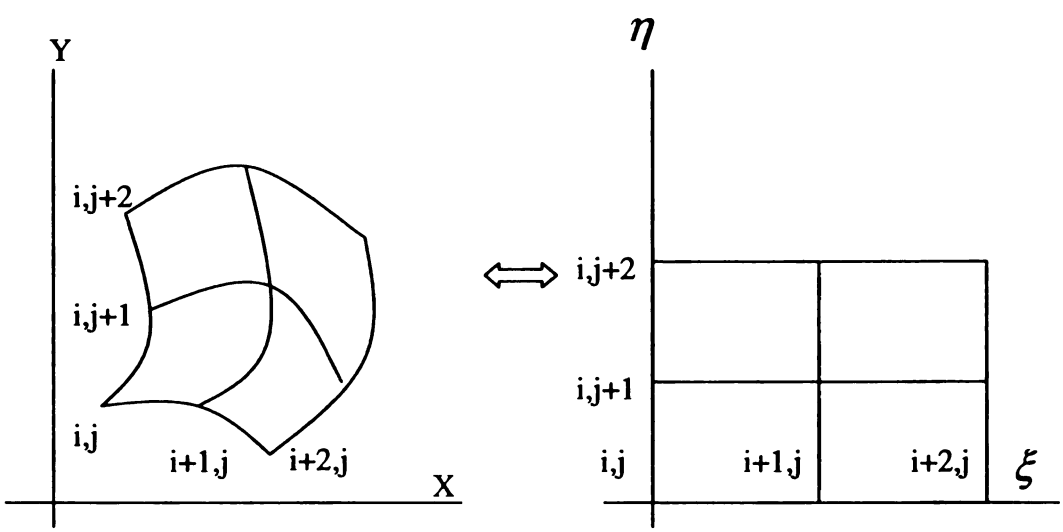

Figure 1. The Coordinate Transformation in 2-D

where X,Y,Z can be expressed as a function of generalized coordinates ξ,η,ζ and vice-versa; i.e.,

$$X = X(\xi,\eta,\zeta) \tag{2.2}$$

$$Y = Y(\xi,\eta,\zeta) \tag{2.3}$$

$$Z = Z(\xi,\eta,\zeta) \tag{2.4}$$

If the Laplacian is used to relate X,Y,Z and ξ,η,ζ, (Ref. 4) then:

$$\nabla^2\xi=0 \quad or \quad \left(\frac{\partial^2}{\partial X^2}+\frac{\partial^2}{\partial Y^2}+\frac{\partial^2}{\partial Z^2}\right)\xi=0 \quad or \quad \xi_{XX}+\xi_{YY}+\xi_{ZZ}=0 \tag{2.5}$$

$$\nabla^2\eta=0 \quad or \quad \left(\frac{\partial^2}{\partial X^2}+\frac{\partial^2}{\partial Y^2}+\frac{\partial^2}{\partial Z^2}\right)\eta=0 \quad or \quad \eta_{XX}+\eta_{YY}+\eta_{ZZ}=0 \tag{2.6}$$

$$\nabla^2\zeta=0 \quad or \quad \left(\frac{\partial^2}{\partial X^2}+\frac{\partial^2}{\partial Y^2}+\frac{\partial^2}{\partial Z^2}\right)\zeta=0 \quad or \quad \zeta_{XX}+\zeta_{YY}+\zeta_{ZZ}=0 \tag{2.7}$$

Step I: Each derivative with respect to the Coordinate variables can be expressed as derivatives of generalized coordinates as illustrated.

$$\frac{\partial}{\partial X}=\xi_X\frac{\partial}{\partial\xi}+\eta_X\frac{\partial}{\partial\eta}+\zeta_X\frac{\partial}{\partial\zeta} \tag{2.8}$$

For convenience we will derive all the formulations for one of the dimensions. Corresponding equations in other dimensions can easily be derived using the similar logic. Proceeding to second derivatives

$$\frac{\partial^2}{\partial X^2} = \frac{\partial}{\partial X}\left(\frac{\partial}{\partial X}\right) = \frac{\partial}{\partial X}\left(\xi_X \frac{\partial}{\partial \xi} + \eta_X \frac{\partial}{\partial \eta} + \zeta_X \frac{\partial}{\partial \zeta}\right) \tag{2.9}$$

Expanding,

$$\frac{\partial^2}{\partial X^2} = \xi_X \xi_X \frac{\partial^2}{\partial \xi^2} + \xi_X \eta_X \frac{\partial^2}{\partial \xi \partial \eta} + \xi_X \zeta_X \frac{\partial^2}{\partial \xi \partial \zeta} + \xi_{XX} \frac{\partial}{\partial \xi} + \eta_X \eta_X \frac{\partial^2}{\partial \eta^2} + \eta_X \xi_X \frac{\partial^2}{\partial \xi \partial \eta} + \eta_X \xi_X \frac{\partial^2}{\partial \partial \eta} + \eta_{XX} \frac{\partial}{\partial \eta} + \zeta_X \zeta_X \frac{\partial^2}{\partial \zeta^2} + \zeta_X \xi_X \frac{\partial^2}{\partial \xi \partial \zeta} + \zeta_X \eta_X \frac{\partial^2}{\partial \zeta \partial \eta} + \zeta_{XX} \frac{\partial}{\partial \zeta} \tag{2.10}$$

Rearranging,

$$\frac{\partial^2}{\partial X^2} = \xi_{xx} \frac{\partial}{\partial \xi} + \eta_{xx} \frac{\partial}{\partial \eta} + \zeta_{xx} \frac{\partial}{\partial \zeta} + 2*(\xi_x \eta_x \frac{\partial^2}{\partial \xi \partial \eta} + \xi_x \zeta_x \frac{\partial^2}{\partial \xi \partial \zeta} + \zeta_x \eta_x \frac{\partial^2}{\partial \zeta \partial \eta}) + \xi_x^2 \frac{\partial^2}{\partial \xi^2} + \eta_x^2 \frac{\partial^2}{\partial \eta^2} + \zeta_x^2 \frac{\partial^2}{\partial \zeta^2} \tag{2.11}$$

Similarly deriving in Y, Z, adding up and using equations 2.5, 2.6, and 2.7,

$$\nabla^2 = \frac{\partial^2}{\partial X^2} + \frac{\partial^2}{\partial Y^2} + \frac{\partial^2}{\partial Z^2} \tag{2.12}$$

$$\nabla^2 = (\xi_X^2 + \xi_Y^2 + \xi_Z^2)\frac{\partial^2}{\partial \xi^2} + (\eta_X^2 + \eta_Y^2 + \eta_Z^2)\frac{\partial^2}{\partial \eta^2} + (\zeta_X^2 + \zeta_Y^2 + \zeta_Z^2)\frac{\partial^2}{\partial \zeta^2} +$$

$$2(\xi_X \eta_X + \xi_Y \eta_Y + \xi_Z \eta_Z)\frac{\partial^2}{\partial \xi \partial \eta} + 2(\eta_X \zeta_X + \zeta_Y \eta_Y + \zeta_Z \eta_Z)\frac{\partial^2}{\partial \zeta \partial \eta} + \qquad (2.13)$$

$$2(\xi_X \zeta_X + \xi_Y \zeta_Y + \xi_Z \zeta_Z)\frac{\partial^2}{\partial \xi \partial \zeta}$$

An important measure of mapping and grid generation is the mathematical entity Jacobian. A positive definite Jacobian indicates the correctness of transformation. The Jacobian, which is the direct measure of physical magnification of transformation, is defined as

$$J = \frac{\partial(X,Y,Z)}{\partial(\xi,\eta,\zeta)} = \begin{vmatrix} X_\xi & X_\eta & X_\zeta \\ Y_\xi & Y_\eta & Y_\zeta \\ Z_\xi & Z_\eta & Z_\zeta \end{vmatrix} \qquad (2.14)$$

i.e.;

$$J = X_\xi (Y_\eta Z_\zeta - Z_\eta Y_\zeta) - X_\eta (Y_\xi Z_\zeta - Y_\zeta Z_\xi) + X_\zeta (Y_\xi Z_\eta - Y_\eta Z_\xi) \qquad (2.15)$$

Derivatives of generalized coordinates can be expressed in terms of Jacobian and the derivatives of cartesian coordinates, e.g.,

$$\xi_x = \frac{1}{J}\begin{vmatrix} 1 & 0 & 0 \\ Y_\xi & Y_\eta & Y_\zeta \\ Z_\xi & Z_\eta & Z_\zeta \end{vmatrix} = \frac{1}{J}(Y_\eta Z_\zeta - Y_\zeta Z_\eta) \tag{2.16}$$

And defining,

$$\alpha_1 = (\xi_X^2 + \xi_Y^2 + \xi_Z^2) \tag{2.17}$$

$$\alpha_2 = (\eta_X^2 + \eta_Y^2 + \eta_Z^2) \tag{2.18}$$

$$\alpha_3 = (\zeta_X^2 + \zeta_Y^2 + \zeta_Z^2) \tag{2.19}$$

$$\beta_1 = \xi_X \eta_X + \xi_Y \eta_Y + \xi_Z \eta_Z \tag{2.20}$$

$$\beta_2 = \zeta_X \eta_X + \zeta_Y \eta_Y + \zeta_Z \eta_Z \tag{2.21}$$

$$\beta_3 = \xi_X \zeta_X + \xi_Y \zeta_Y + \xi_Z \zeta_Z \tag{2.22}$$

The equation (2.13) reduces to,

$$\nabla^2 = \alpha_1 \frac{\partial^2}{\partial \xi^2} + \alpha_2 \frac{\partial^2}{\partial \eta^2} + \alpha_3 \frac{\partial^2}{\partial \zeta^2} + 2\beta_1 \frac{\partial^2}{\partial \xi \partial \eta} + 2\beta_2 \frac{\partial^2}{\partial \zeta \partial \eta} + 2\beta_3 \frac{\partial^2}{\partial \xi \partial \zeta} \tag{2.23}$$

Step II: Interchange dependent and independent variables

$$\left(\frac{\partial^2}{\partial X^2} + \frac{\partial^2}{\partial Y^2} + \frac{\partial^2}{\partial Z^2} \right) \xi = 0 \quad \Leftrightarrow$$
$$\left(\alpha_1 \frac{\partial^2}{\partial \xi^2} + \alpha_2 \frac{\partial^2}{\partial \eta^2} + \alpha_3 \frac{\partial^2}{\partial \zeta^2} + 2\beta_1 \frac{\partial^2}{\partial \xi \partial \eta} + 2\beta_2 \frac{\partial^2}{\partial \zeta \partial \eta} + 2\beta_3 \frac{\partial^2}{\partial \xi \partial \zeta} \right) X = 0$$

(2.24)

Equation (2.24) represents the two-coupled PDEs that must be solved. To solve equation (2.24) in pseudo time, the formulation changes to

$$\frac{\partial X}{\partial \tau} = \left(\alpha_1 \frac{\partial^2}{\partial \xi^2} + \alpha_2 \frac{\partial^2}{\partial \eta^2} + \alpha_3 \frac{\partial^2}{\partial \zeta^2} + 2\beta_1 \frac{\partial^2}{\partial \xi \partial \eta} + 2\beta_2 \frac{\partial^2}{\partial \zeta \partial \eta} + 2\beta_3 \frac{\partial^2}{\partial \xi \partial \zeta} \right) X \tag{2.25}$$

Using finite differencing,

$$\frac{X^{n+1} - X^n}{\Delta \tau} = \alpha_1^n \frac{X^n_{i+1,j,k} - 2X^n_{i,j,k} + X^n_{i-1,j,k}}{\Delta \xi^2} +$$
$$\alpha_2^n \frac{X^n_{i,j+1,k} - 2X^n_{i,j,k} + X^n_{i,j-1,k}}{\Delta \eta^2} + \alpha_3^n \frac{X^n_{i,j,k+1} - 2X^n_{i,j,k} + X^n_{i,j,k-1}}{\Delta \zeta^2} +$$
$$2\beta_1^n (ux)^n_{i,j,k} + 2\beta_2^n (vx)^n_{i,j,k} + 2\beta_3^n (wx)^n_{i,j,k}$$

(2.26)

$$\beta_1 = -\frac{1}{J^2}((Y_\eta Z_\zeta - Y_\zeta Z_\eta)(Y_\xi Z_\zeta - Y_\zeta Z_\xi) + (X_\eta Z_\zeta - X_\zeta Z_\eta)(X_\xi Z_\zeta - X_\zeta Z_\xi) + (X_\eta Y_\zeta - X_\zeta Y_\eta)(X_\xi Y_\zeta - X_\zeta Y_\xi)) \tag{2.27}$$

and,

$$J = X_\xi(Y_\eta Z_\zeta - Z_\eta Y_\zeta) - X_\eta(Y_\xi Z_\zeta - Y_\zeta Z_\xi) + X_\zeta(Y_\xi Z_\eta - Y_\eta Z_\xi) \tag{2.28}$$

the derivatives of cartesian coordinates can be expressed using central differencing,

$$X_\xi = \frac{X_{i+1,j,k} - X_{i-1,j,k}}{2\Delta\xi} \tag{2.29}$$

2.3 The Generalized Laplace operator for unstructured mesh

The laplace operator formulated for three-dimensional single-block structured mesh can be generalized for unstructured mesh. The generalized laplacian is based on another form of discretized laplacian where each point inside the region must be an average of the neighboring points. This can mathematically be represented as:

$$\delta x = (\sum_{i=1}^{n} X_i - nX)/n \tag{2.30}$$

where δx is the required shift in x coordinate of point X, and X_i represents x coordinate of the neighboring points. This simple averaging operator works well in cases where the

unstructured cells are largely isotropic (e.g., the angles in a triangular mesh are all close to 60 degrees). In case where this is not true we will get the desired results by using weighted averaging instead of simple averaging. This averaging can be based on nodal distance or face area.

2.4 The Poisson operator

The poisson operator is used to cluster the grid points in a particular vectorial direction in a particular region of interest. The general poisson operator is represented by:

$$\frac{\partial^2 v(\xi,\eta,\varsigma)}{\partial \xi^2}+\frac{\partial^2 v(\xi,\eta,\varsigma)}{\partial \eta^2}+\frac{\partial^2 v(\xi,\eta,\varsigma)}{\partial \varsigma^2}+s(\xi,\eta,\varsigma)=0 \tag{2.31}$$

The general poisson operator does not guarantee that there would be no overlap of grid and depending on the choice of function s(X,Y,Z) might assume different forms. In case when s(X,Y,Z) = 0, the poisson's equation reduces to laplace equation. Simplifying to one dimension the generalized poisson equation reduces to:

$$\frac{\partial^2 v(\xi,\eta,\varsigma)}{\partial \xi^2}+s(\xi,\eta,\varsigma)=0 \tag{2.32}$$

A clever choice of the function $s(\xi,\eta,\varsigma)$ would guarantee no overlap of grids. A simplified case is:

$$\frac{\partial^2 X}{\partial \xi^2}+k\frac{\partial X}{\partial \xi}=0 \tag{2.33}$$

where, $0 < k < 2$. This clusters the grid in x-direction with guarantee of no overlap. We would use this simplified case to demonstrate a few grid-clustering results.

CHAPTER 3 GRID QUALITY MEASURES

3.1 Introduction

As discussed in Chapter 1, smoothing and clustering are some of the tools to create an "acceptable" grid from a pre-existing grid. The next question that arises here is what exactly an "acceptable" grid is. What mathematical or other problem specific scales (e.g., based on flow physics) can be defined to measure the grid quality and what do they mean? These measures are an objective tool in our hand that helps us identify the "good" region from the "bad" region. Apart from just being a mathematical measure it physically identifies the possible points in space the solution might potentially fail. This gives us the freedom to apply the smoothing and clustering tools only in areas of interest where the grid quality is "bad" or "unacceptable" or where it is crucial to resolve the flow physics very minutely. In other words we localize the problem before we fix it. This approach thus is computationally more efficient.

Many grid quality measures have been developed for flow problems. Some measures are purely mathematical and based of parameters of the grid or cell, while some are based on flow solution or flow physics. To simplify the analysis we have used a couple of basic mathematical measures in this analysis. These are smoothness, aspect ratio, and skewness. These grid quality measures are simple to calculate and easy to implement. These measures might exist in some variations but the basic idea remains the same. The measures objectively segregate the poor quality region for the whole domain. This information helps us in selectively implementing grid relaxation, refinement and error analysis.

On the same lines, the Laplacian operator developed is not only capable of global grid improvement but also to improve a small portion of the grid, which is called the "dirty" region. The automatic identification and application of the Laplacian operator requires that we extract the bad grid region based on our mathematic definition of "dirty" region using the pre-defined grid quality measures.

3.2 Grid Smoothness

In two-dimensional grids 'Area ratio' at a cell is defined as the ratio of cell face area to adjacent cells (usually four). We can represent the ratio for the cell with the largest or smallest value of the ratio.

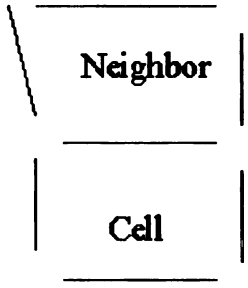

Figure 2. The two-dimensional cell and the top neighbor

$$\text{Area Ratio} = \text{Area (Cell)/Area (Neighbor)} \tag{3.1}$$

An acceptable value of this measure is close to 1. Let us try to understand how this is related to flow physics. For solving a partial differential equation, which is the Navier Stokes equation in our case using finite difference the solution, will always break down at a bad point in case of forward or backward differencing. Central differencing will produce an acceptable result for a single point but if the grid is continuously bad then neighboring values will not correspond and the scheme will break down. This effect is

most dominant close to the boundary layer where we implement the gradient conditions. A sudden jump in the 'Area ratio' breaks down this implementation of the boundary condition.

In three-dimensional grids the 'Volume Ratio' is the measure of smoothness, and is defined as the ratio of cell volume to volume of the neighboring cells. Again we represent the 'Volume Ratio' by a single number that is representative of the worst ratio.

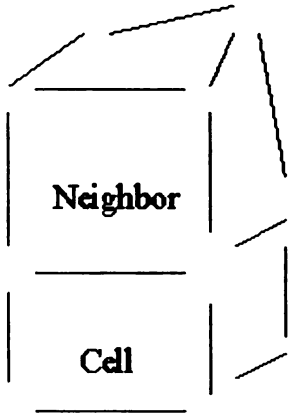

Figure 3. Three-dimensional cell and the top neighbor

$$\text{Volume Ratio} = \text{Volume (Cell)}/\text{Volume (Neighbor)} \tag{3.2}$$

3.3 Grid Aspect Ratio

In a two-dimensional grid the 'Aspect ratio' of a cell is defined as the ratio of cell width and the cell height.

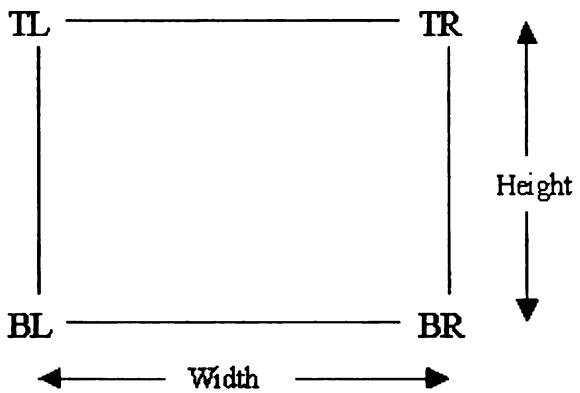

Figure 4. Two-dimensional cell

$$\text{Aspect Ratio} = \text{Cell Width} / \text{Cell Height} \tag{3.3}$$

In three-dimensional grids the 'Aspect Ratio' is defined as the ratio of maximum face area to minimum face area in the cell.

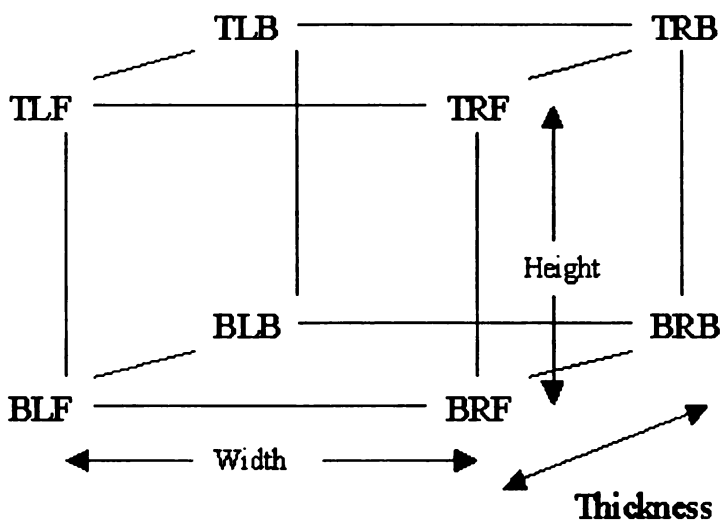

Figure 5. Three-dimensional cell

$$\text{Aspect Ratio} = \text{Maximum Area of a face} / \text{Minimum Area of a face} \tag{3.4}$$

For a highly irregular flow where the direction of flow is not easy to discern, it is advisable to keep the 'Aspect Ratio' as close to 1 as possible. In many cases though high

aspect ratio grids are intentionally used. Let's consider the case of fully developed laminar boundary layer in flow through a pipe or a channel. In this case there is no significant change in the flow properties in the flow direction. In other words the gradient in the flow properties in the direction of the flow is negligible. The gradient is still significant in the direction perpendicular to the flow. In this case we can utilize grids with high aspect ratio to increase computational efficiency. We must exercise caution when making such a choice. In highly turbulent flows with recirculation and other complexities like combustion the aspect ration must be kept close to 1.

3.4 Grid Skewness

In two-dimensional grids the 'Skewness' is defined as the ratio of the two principal diagonals of the cell.

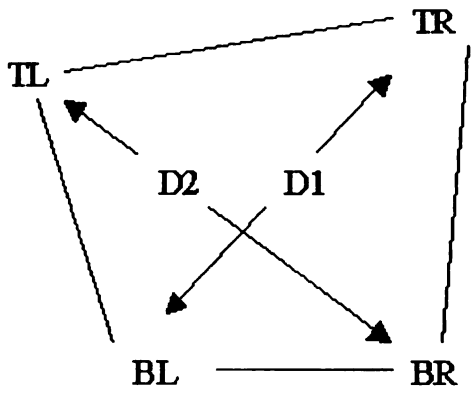

Figure 6. Two-dimensional cell with diagonals

$$\text{Skewness} = D2 / D1 \tag{3.5}$$

In three-dimensional grids the 'skewness' is defined as the ratio of largest principal diagonal to smallest principal diagonal.

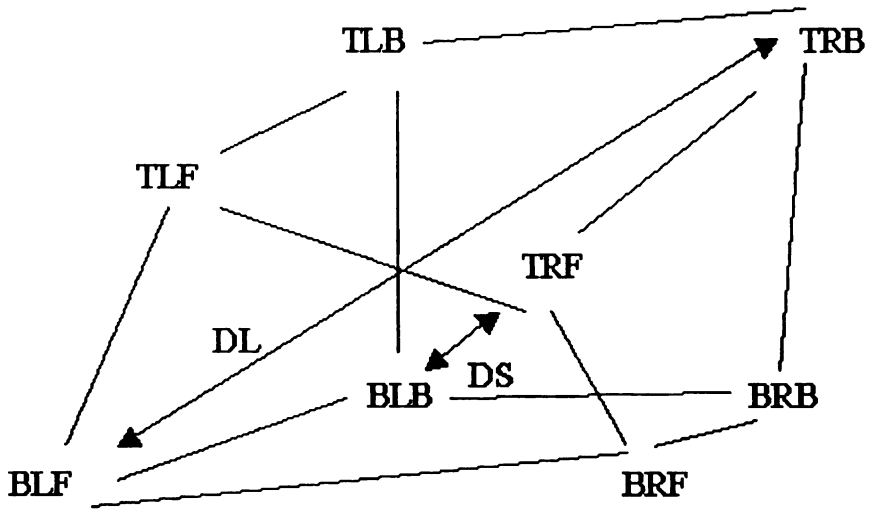

Figure 7. Three-dimensional cell with diagonals

$$\text{Skewness} = DL / DS \tag{3.6}$$

If the grid is generated using advancing front schemes grid of high skewness is usually produced at the surface. Sometimes when the surfaces intersect at oblique angles, grids of high skewness are produced. Usually skewness between 0.90 and 1 are universally acceptable for most practical purposes though different applications have different requirements. For unstructured grids skewness is defined as the maximum angle that a face normal deviates from the vector between the node of the tetrahedron not on the face and the centroid of the face. In this case, a value close to zero indicates an equilateral tetrahedron and is the desirable case.

CHAPTER 4 CODE STRUCTURE

4.1 Introduction

The input to the system is a multi block structured or unstructured grid in an unstructured data format. This data format is the simplest data format that can be easily read from and written to a flat file. The computational time is proportional to the grid size and reading a file is computationally most expensive. This is because the file is read to sequentially extract the grid information like the neighborhood and connectivity data. The process is improved by using a cache of this transient data during the analysis and a sequential file is generated at the end that can be used for subsequent operation on the same grid file. This improves the computational efficiency of the code. A translator has been used in each step to format the grid according to the visualization tool used at each user interface. This will be reduced down in case a standardized tool or format is followed. The algorithms discussed in Chapter 2 have been implemented using C++ and Fortran. The basic code skeleton has been explained in the following paragraphs and flow charts.

4.2 The Algorithm

4.2.1 Decipher the Grid Structure

The formulation has been developed independent of the format in which data is available. The grid can directly or indirectly address the neighborhood issue. In former case all the points (nodes) in the domain can be uniquely identified with a three dimensional index. Laplace equation can be directly solved in the domain using explicit pointers to the nodes. In later case the most commonly used format is sometimes referred to as 'Block-Structure' or FE Block structure. This format enumerates all the points and first specifies the point locations and then specifies the connectivity (tri, quad, tet or hex) of the

enumerated points. The indirect addressing of neighboring points requires further processing to extract enough neighborhood information for each point to be able to solve the finite difference formulation.

4.2.2 Extract the Neighborhood Information

The understanding of the grid structure makes it feasible to extract, remove or introduce new points in the domain and to rearrange the existing points in the domain. The extraction of neighborhood information is the first step in the algorithm. As mentioned earlier, we only know all the points in the domain, which are enumerated, and secondly how they are connected to each other. The neighborhood information can be obtained from this information. This makes it possible to implement the averaging Laplace strategy.

4.2.3 Identify the Bad Quality Region

After extracting the neighborhood information for each node, we can proceed to measure the grid quality based on a suitable measure. This indicates the dirty or bad quality region in the flow domain. We now would want to improve the grid quality or repair the bad grid using our algorithm. In two approaches we can either fix an independent boundary in the sense that the nodes on the boundary do not move only the interior is rearranged or make the nodes on the boundary free to slide on the boundary. The choice depends on application, geometry information available as well as on personal preference.

4.2.4 Refine / Coarsen the Grid

For further grid refinement or grid generation we introduce new points in the domain. The new points can be introduced as per Delaunay refinement strategy or we can just introduce a new point over an existing point. This way it is easier to control exact number

of new cells we want to introduce. We can introduce as few as one more cell. Similarly the grid can be easily coarsened in a required area by deleting some nodes.

4.2.5 Relax the Grid

Now apply the laplace operator in the whole or a part of the domain. The laplace operator ascertains that there is no overlap in the grids and that grids increasingly become convex. In the process the connectivity information remains the same but the point coordinates change. After enough iterations or attaining the convergence we get the final location of points. This overwrites the input point coordinate information.

4.3 Program Flow-Chart

The program outline has been put in a flow chart to demonstrate the logic and data flow in the program. Figure.4.1 elaborates the general skeleton of the program and Figures.4.2 and 4.3 elaborate the basic steps inside the global and local operator.

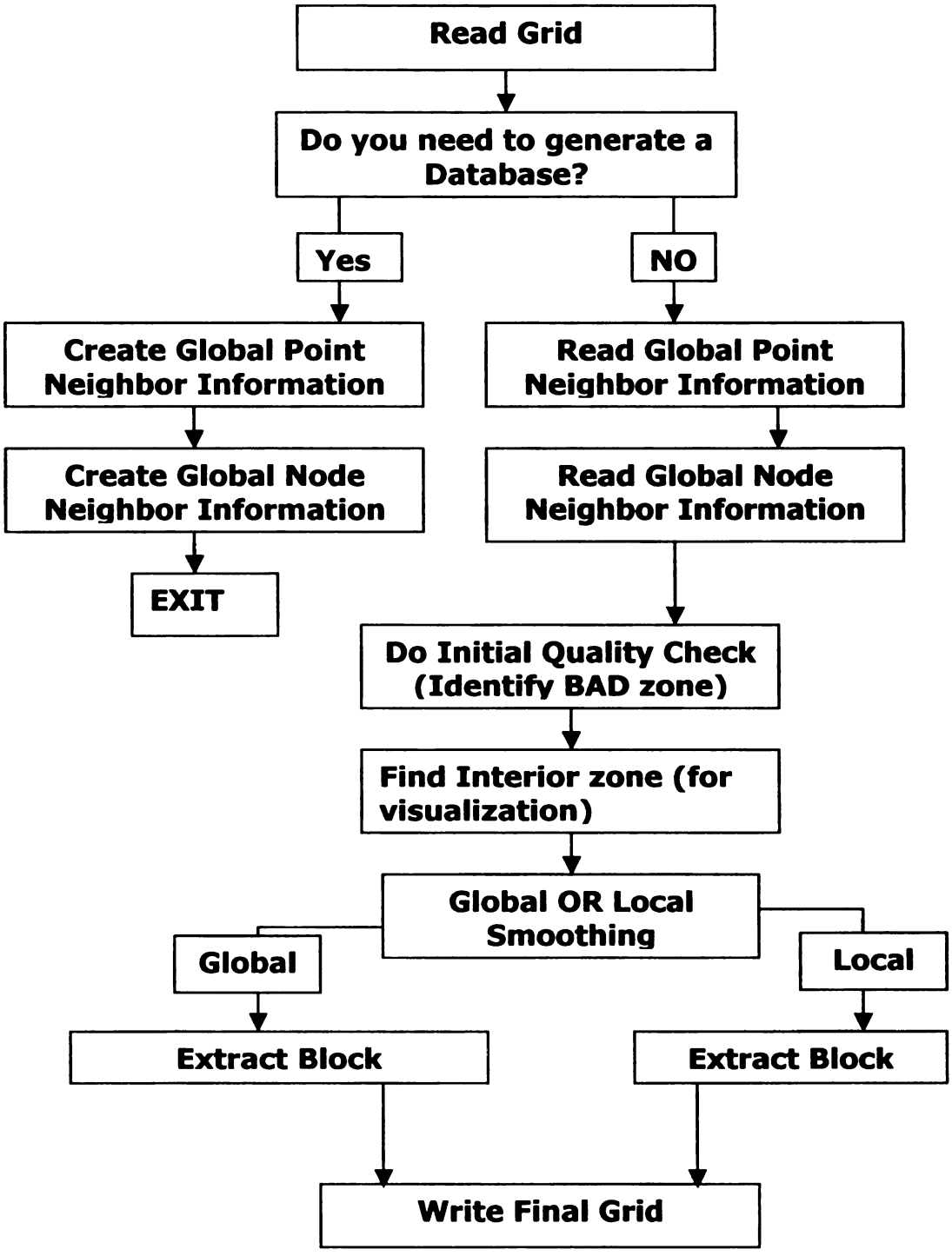

Figure 8. The Program Flow Chart

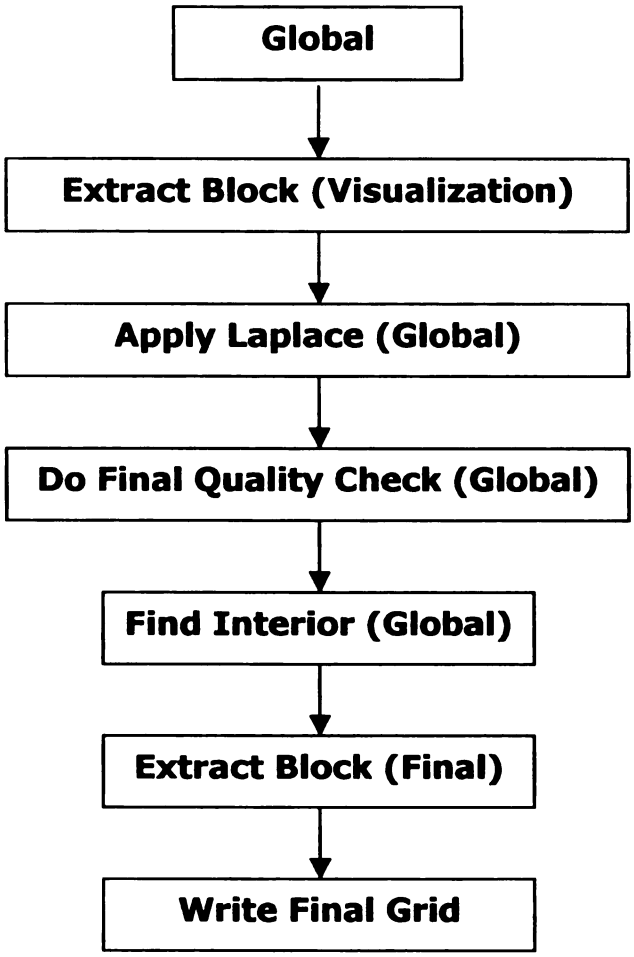

Figure 9. The Global PDE Operator Flow Chart

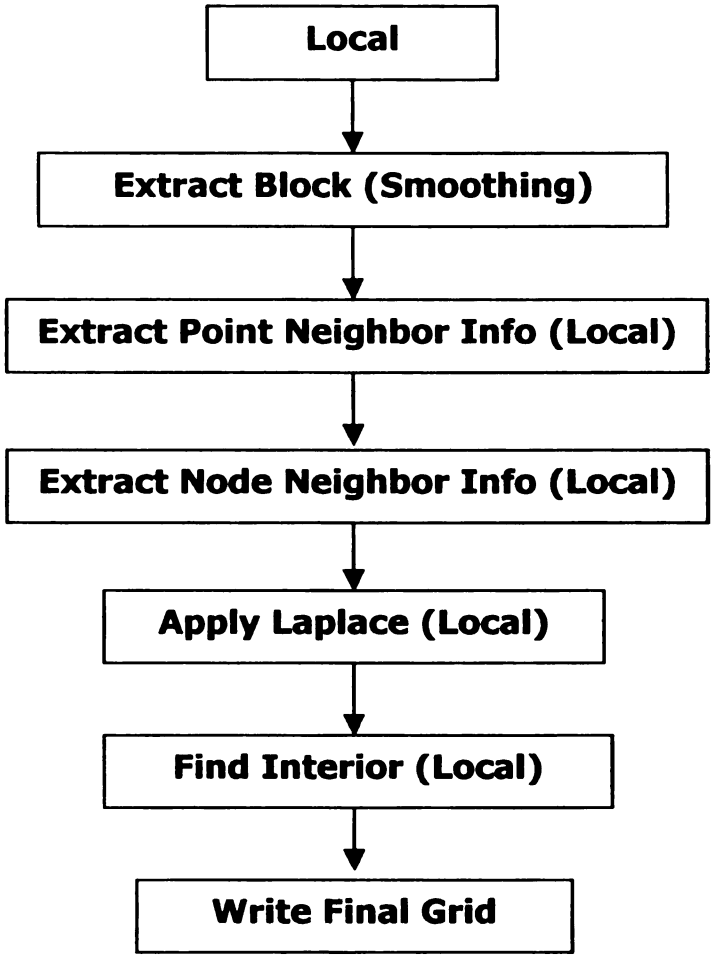

Figure 10. The Local PDE Operator Flow Chart

CHAPTER 5 RESULTS AND DISCUSSION

The algorithms developed have been applied to various types of grids used in different applications. Some of the representative results are shown and discussed in this chapter. The Laplace and Poisson operators have been implemented and applied to structured and unstructured grid systems.

The Laplacian algorithm is capable of performing grid relaxation without any overlap in grid lines. The Poisson operator can cluster the grids in a chosen direction. The outcome depends on the specific form of the passion equation. In certain cases (as implemented here) it is possible to cluster without overlap. All the algorithms can be applied locally or globally. In former case a bad quality region is extracted from the whole domain and the elliptic operator is applied only to the selected region. In later case the operator is applied on the whole domain. All the approaches, relaxation, refinement, and clustering can be either boundary conforming or sliding. In a boundary conforming grid the nodes on the boundary are fixed and do not move during the iteration. In sliding mesh the boundary nodes are allowed to slide on the boundary surface, making it a more efficient approach in some cases. One restriction though is that the boundary should be mathematically well defined and appropriate interpolation functions should be used. Results demonstrating the capability of the code have been presented in this chapter.

5.1 Multi Block Structured Hexahedral Mesh

The algorithm was particularly applied to grid generated for analysis of flow in an internal combustion engine. The initial grid was generated for analysis of flow using KIVA 3V code. The grid quality is a critical factor in case of analysis using KIVA 3V. The code typically failed to execute (it rejects the input grid) in critical cases where the

piston was close to the top dead center. In this particular application the grid that is attached to the piston moves with the piston in next time step. The rest of the grid moves in a particular quake motion. The movement of a particular grid point depends on its distance from the piston and has no control on actual grid distribution. Relaxation and clustering operators are needed to improve the grid quality. The results shown below (Fig. 10 - Fig. 13) demonstrate a small portion of the grid before and after the operator has been applied. As we can see the final grid is relaxed and has improved smoothness. The improvement has been mathematically measured but in all cases (as ones shown below) the difference can be noticed by visual comparison. The visualization tool used is Tecplot 9.0.

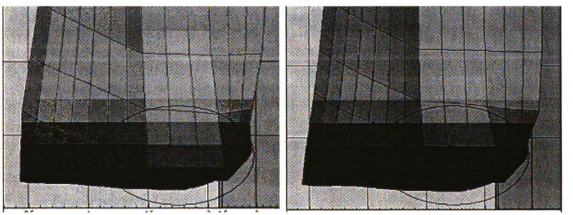

Figure 11. Example of Local grid smoothing with fixed boundary

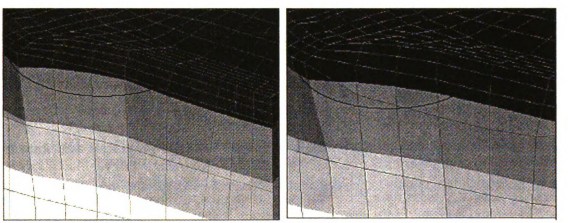

Figure 12. Example of Local grid smoothing with fixed boundary

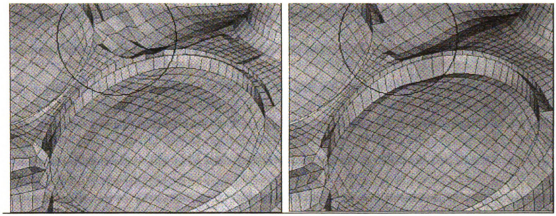

Figure 13. Example of Global grid smoothing with fixed boundary

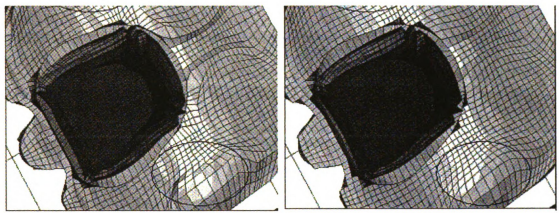

Figure 14. Example of Global grid smoothing with fixed boundary

5.2 Unstructured (triangular) Mesh

5.2.1 Laplace Operator on sliding mesh

The Laplace and Poisson operators were generalized for unstructured meshes. These operators can be applied to triangular and tetrahedral grids. The code capabilities are mostly the same. The operator can be applied across the block boundary, can be implemented locally or globally, and can be applied to sliding boundaries. Some representative results with two-dimensional triangular grid have been presented in the following discussion.

The Laplace operator smoothens the grid or generates cells of fairly equal sizes. This mathematically has been described and measured as area or volume ratios. We start with a simple two-dimensional unstructured grid as shown in Fig.14. We compare two cases that have been implemented. Fig. 15 represents the final grid where the boundary nodes are fixed, that is the grid is not allowed to move or slide on the boundary. Fig. 16 represents the final grid when boundary grid nodes are allowed to slide along the boundary. As we see the sliding mesh produces better results but the application can be limited in cases where the boundary is not well defined.

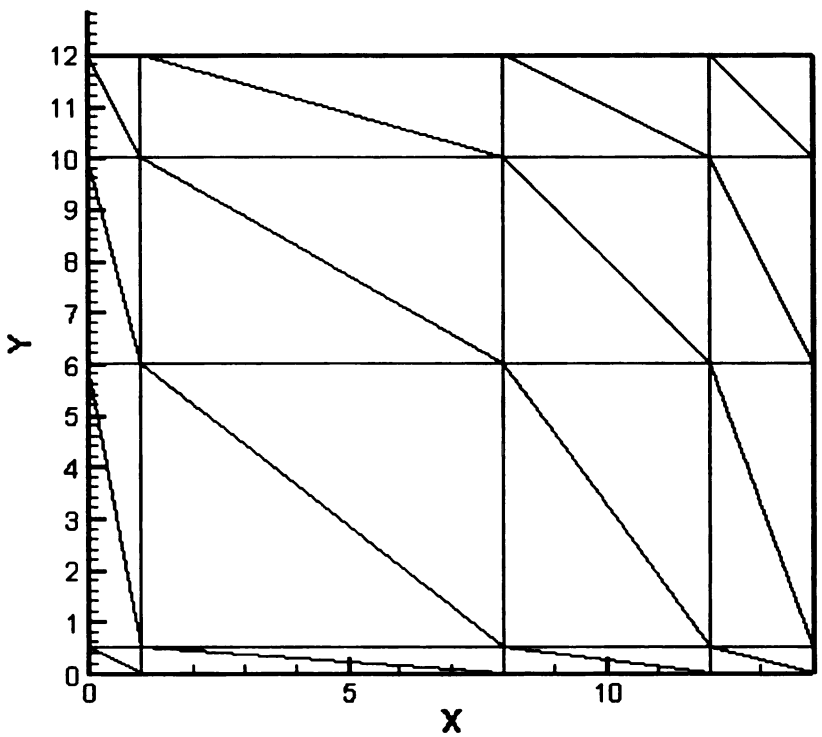

Figure 15. Laplace operator; Initial Bad Triangular Mesh

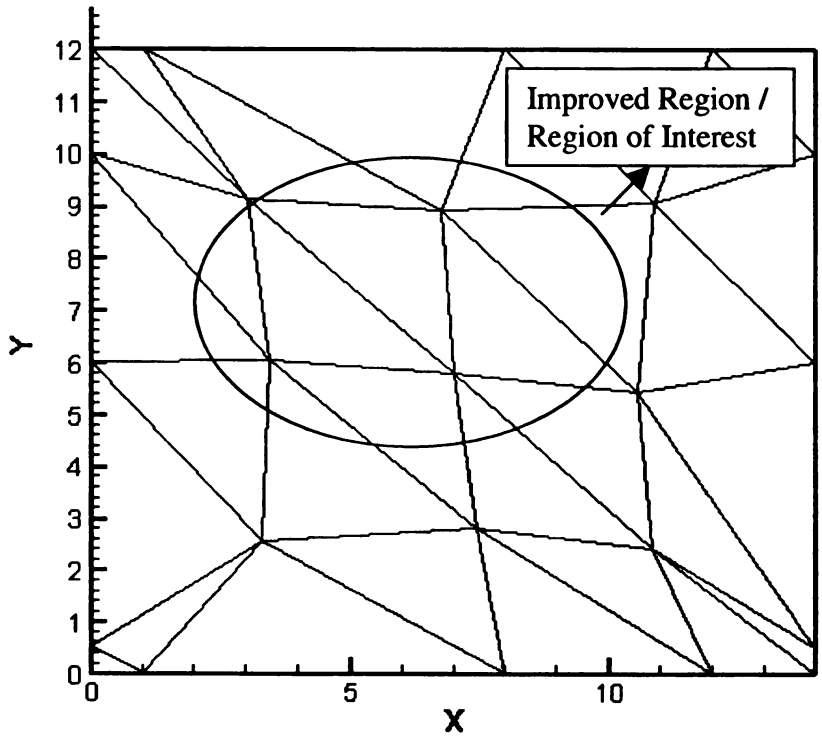

Figure 16. Laplace operator; Boundary Conforming Triangular Mesh, After 1 Iteration

The Internal region is smoothened. Boundary regions still have some highly anisotropic triangular elements. The region of interest is the internal region because the smoothing algorithm in this case is 'boundary confirming' i.e. points on the boundary are not modified. Below shown is result with sliding mesh. Grid quality is improved in the whole domain.

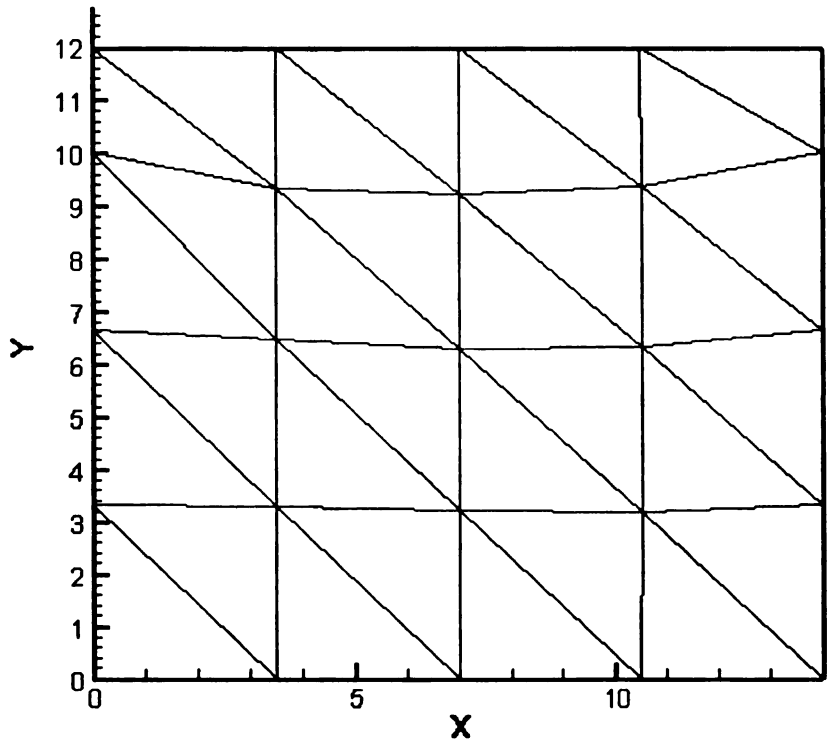

Figure 17. Laplace operator; Sliding Triangular Mesh, After 1 Iteration

5.2.2 Elliptic Operator as a Grid Generation Tool

The advantage of sliding approach is that the Elliptic Operator can be used not only for Grid Relaxation but also very efficiently for Grid Generation and Grid Refinement. The Grid Generation algorithm works exactly the same way as Grid Relaxation algorithm but all the non-boundary nodes are put at the local origin for the grid domain. A simple example is shown below which finally generates the same grid as previous result (after enough iterations). Another faster approach would be that instead of putting all non-boundary points at origin or at a single point on boundary the nodes can be distributed on the boundary.

The Grid Refinement algorithm would again work the same way. In the places / regions that need greater concentration of points we will introduce more points

overlapping the existing points close to the region or at the boundary or at the internal points in the region.

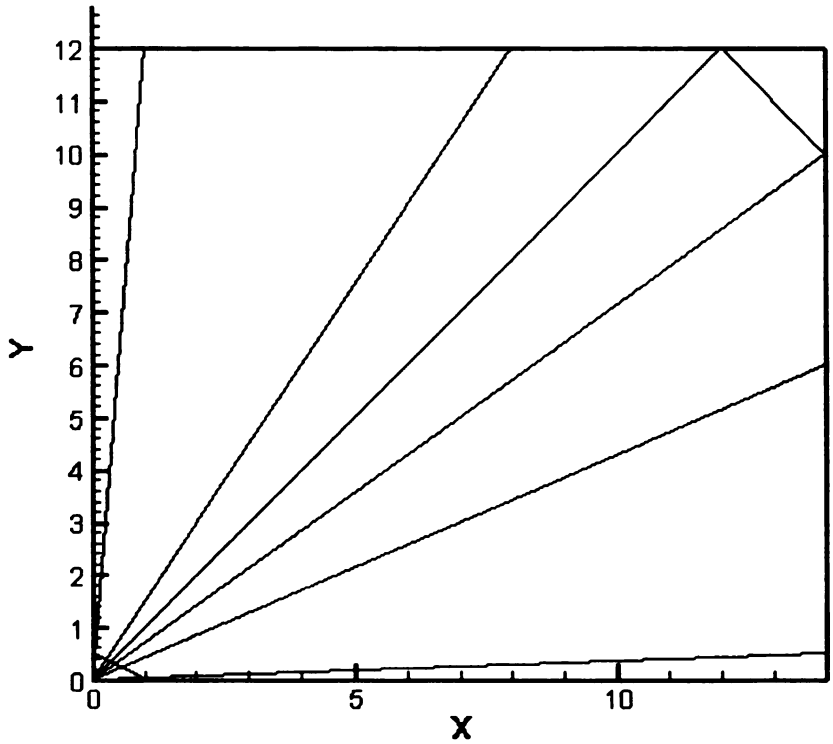

Figure 18. Grid Generation; Initial Mesh

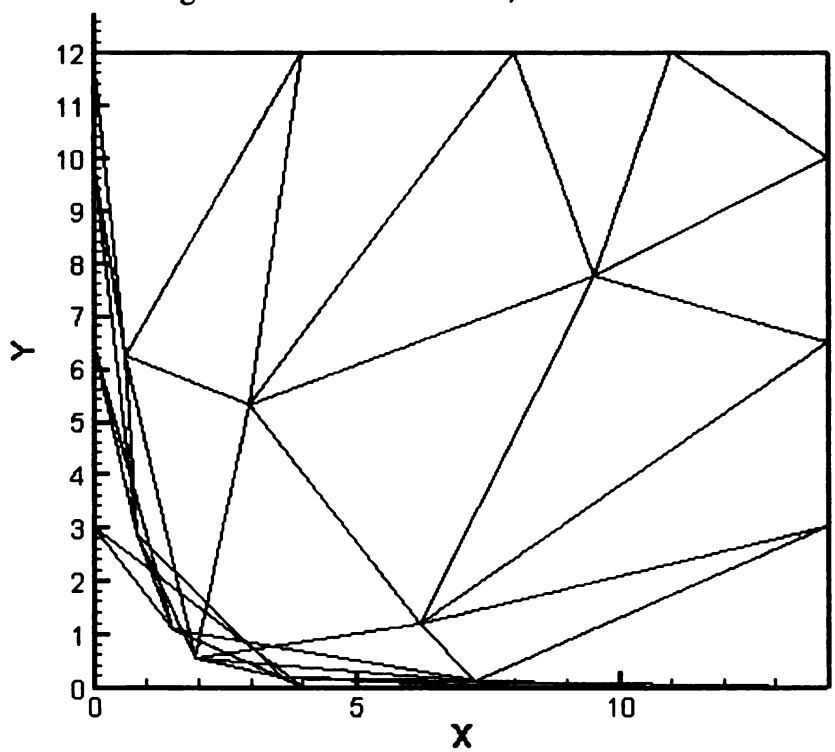

Figure 19. Grid Generation; Mesh after 1 Iteration

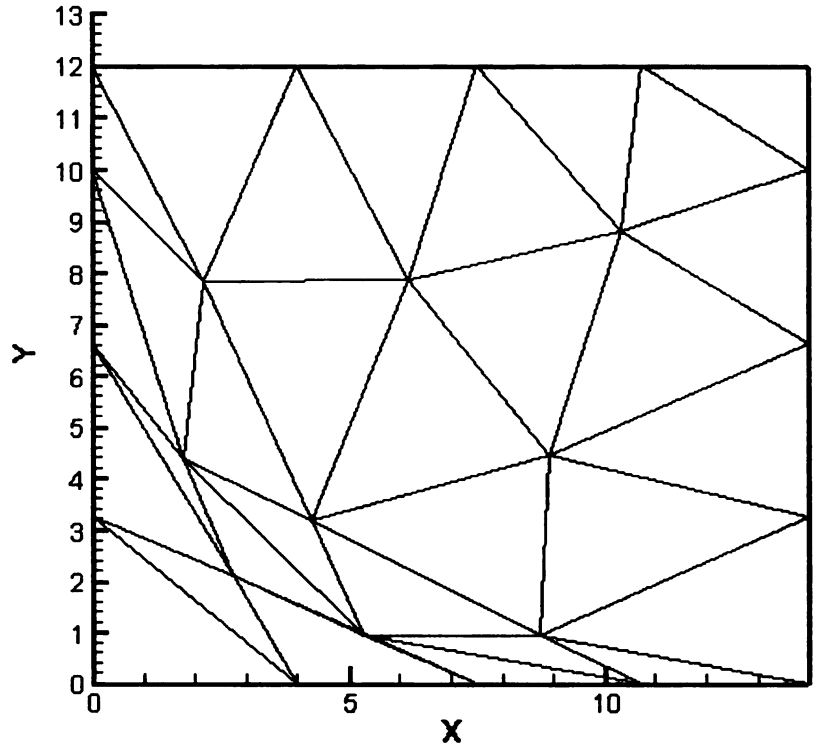

Figure 20. Grid Generation; Mesh after 2 Iterations

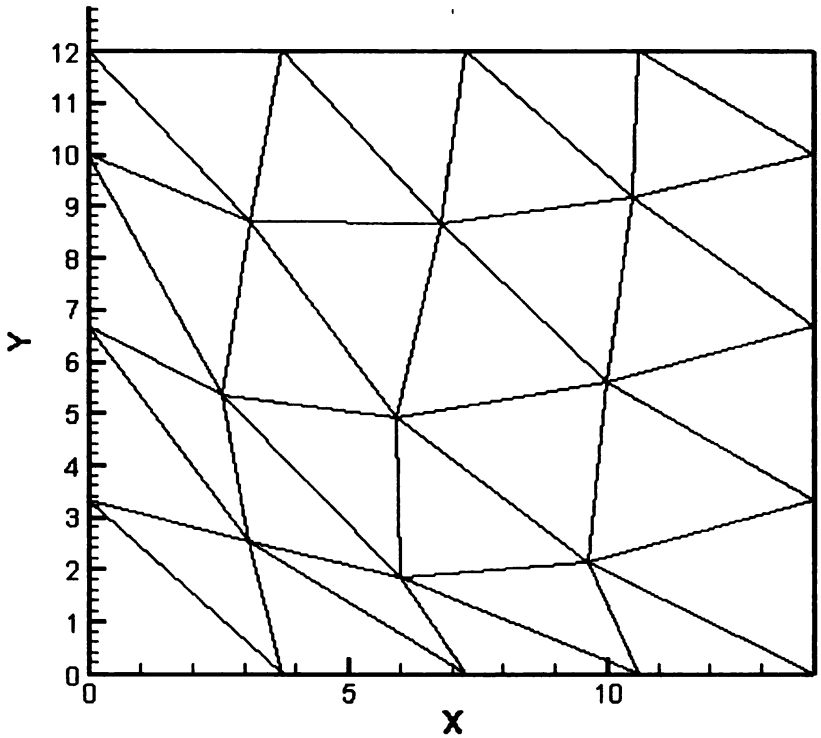

Figure 21. Grid Generation; Mesh after 3 Iterations

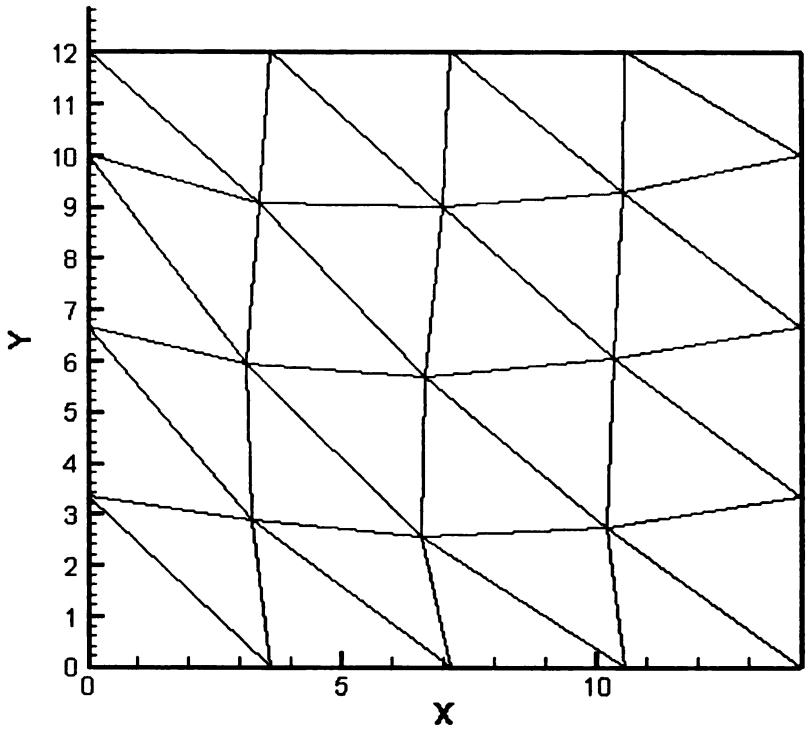

Figure 22. Grid Generation; Mesh after 4 Iterations

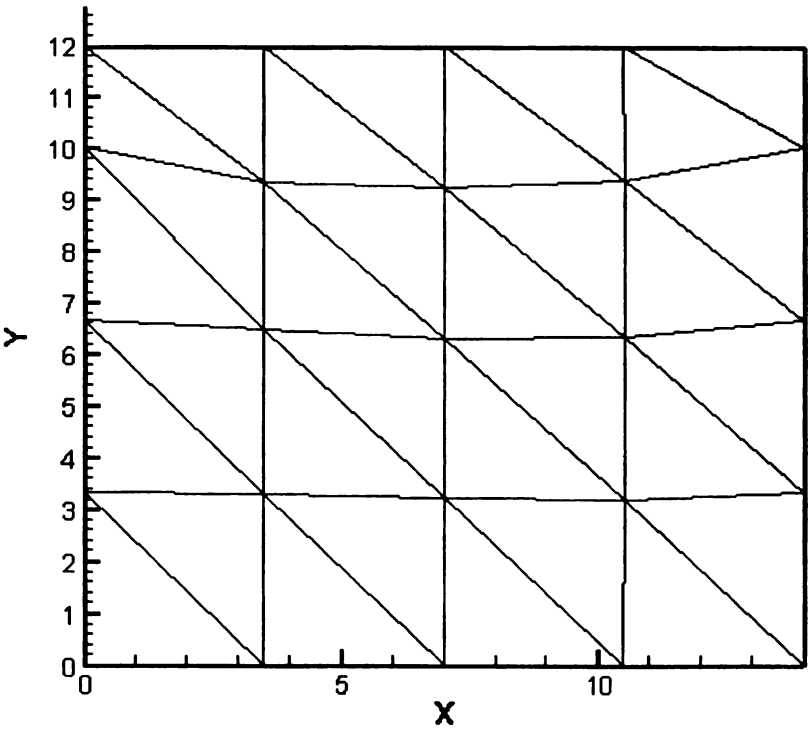

Figure 23. Grid Generation; Mesh after 10 Iterations

5.2.3 Poisson Operator

The Poisson operator has also been generalized to cluster the grid around a point, a line or in a desired direction. A simple example has been demonstrated below.

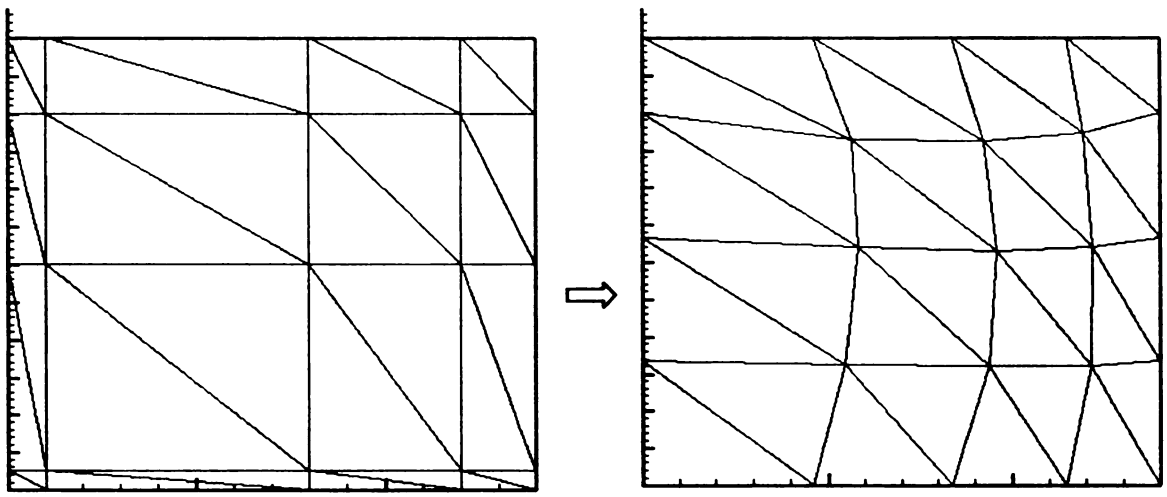

Figure 24. Grid Clustering; Initial and Intermediate grid

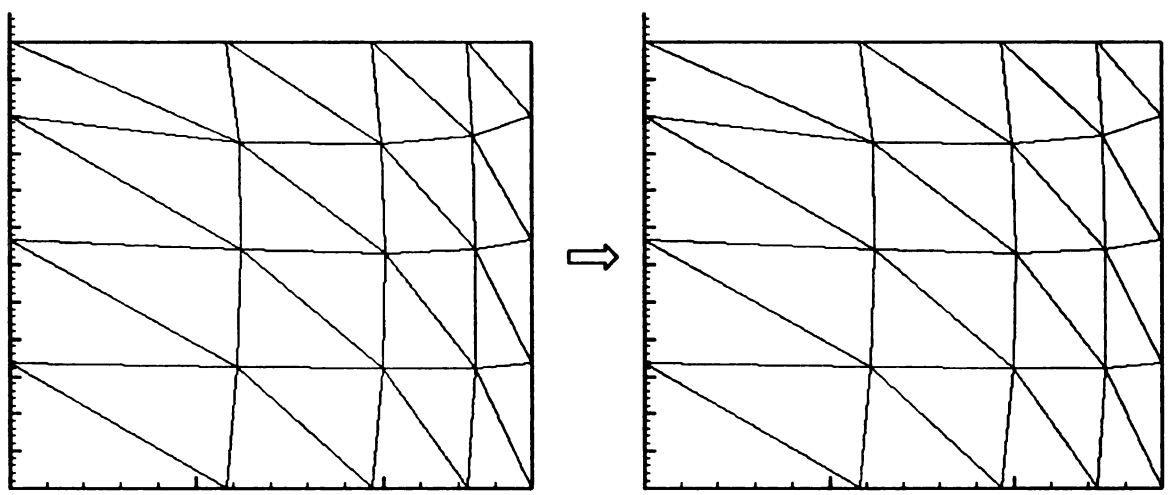

Figure 25. Grid Clustering; Intermediate and Final grid

5.2.4 Grid Refining

Inserting a point in the desired region and applying the elliptic operator in steps can easily achieve grid refining.

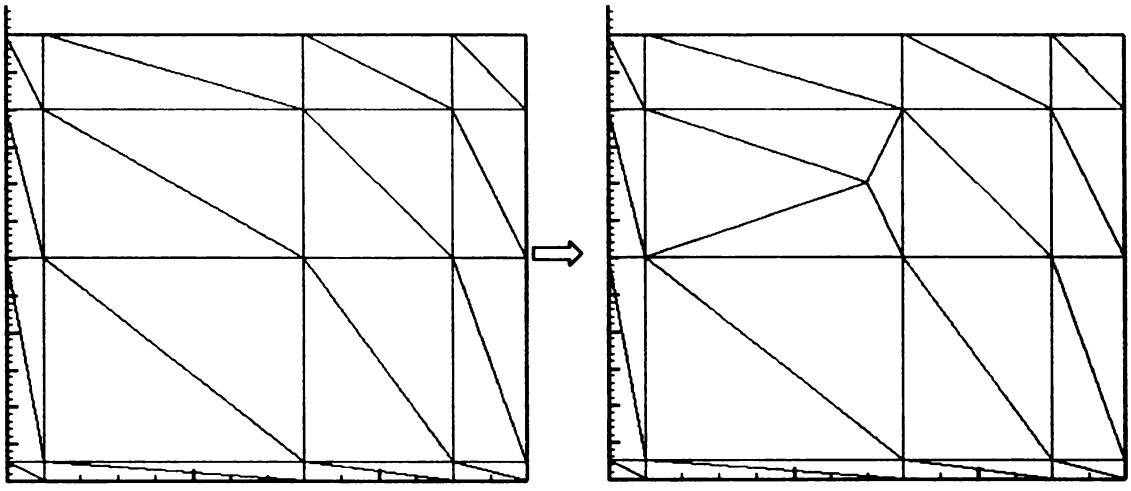

Figure 26. Grid Refining; Initial and Intermediate grid

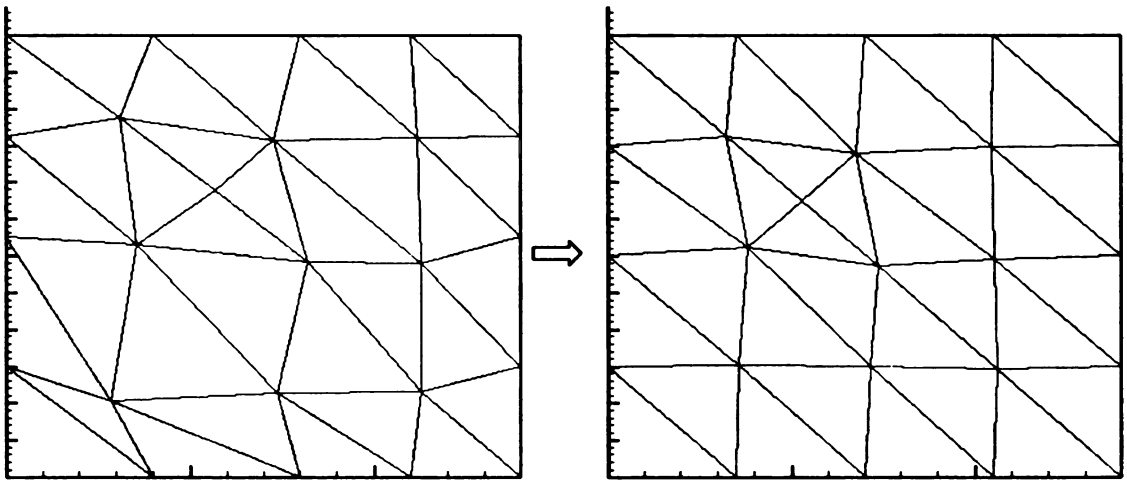

Figure 27. Grid Refining; Intermediate and Final grid

5.2.5 Grid Coarsening

Grid coarsening works same as refining and can be easily achieved by deleting a point in the desired region and applying the elliptic operator in steps.

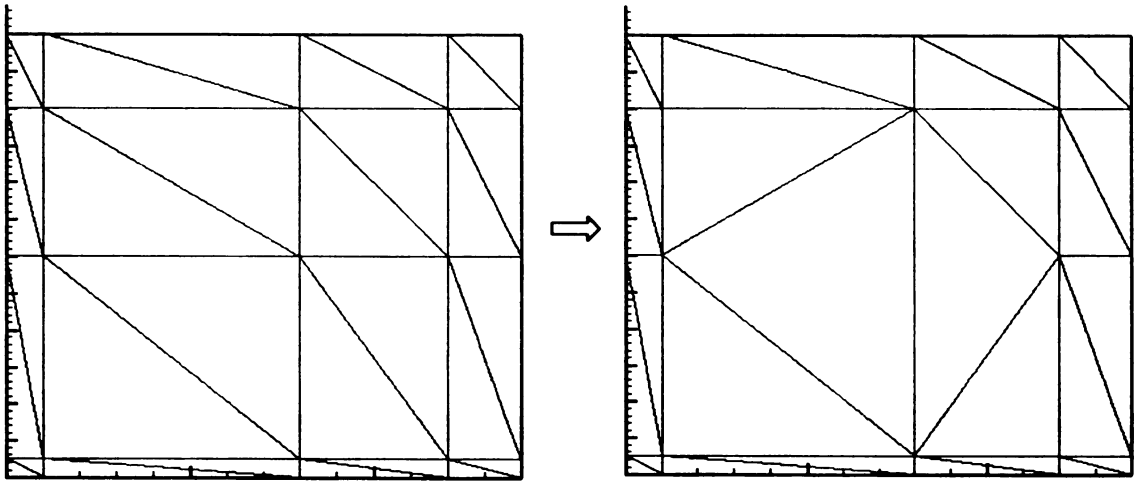

Figure 28. Grid Coarsening; Initial and Intermediate grid

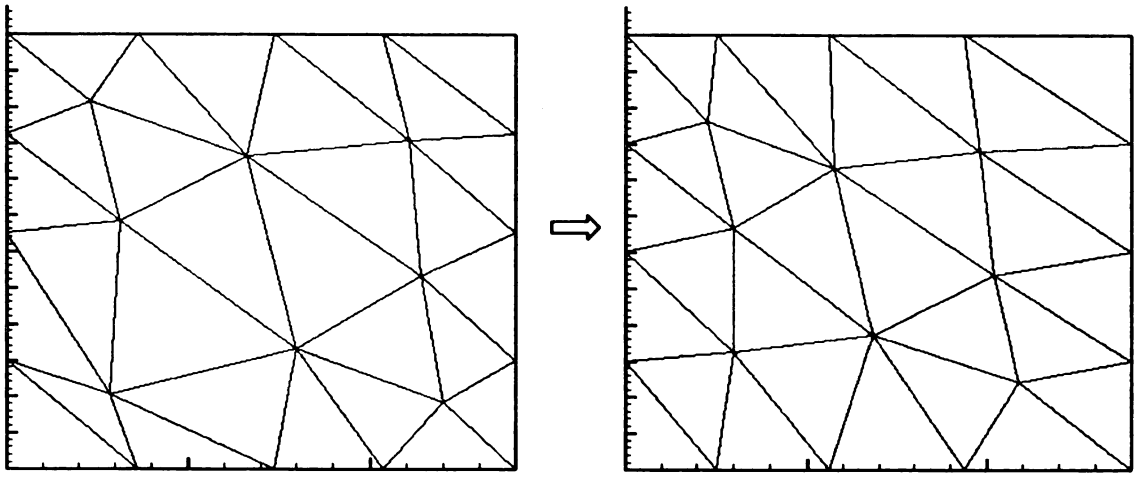

Figure 29. Grid Coarsening; Intermediate and Final grid

REFERENCES

REFERENCES

1. P.R. Eiseman and G. Erlbacher, 'Grid generation for the solution of partial differential equations', ICASE Report No. 87-57, NASA Langley Research Center, Hampton, VA, 1987; also NASA CR-178365, 1987.

2. Shih, T.I-P., Bailey, R.T., Nguyen, H.L., and Roelke, R.J., "Algebraic Grid Generation for Complex Geometries," *International Journal for Numerical Methods in Fluids*, Vol. 13, 1991, pp. 1-31.

3. Steinthorsson, E., Shih, T.I-P., and Roelke, R.J., "Enhancing Control of Grid Distribution in Algebraic Grid Generation," *International Journal for Numerical Methods in Fluids*, Vol. 15, 1992, pp. 297-311.

4. Thompson, J.F., Soni, B.K., and Weatherill, N.P., Editors, *Handbook of Grid Generation*, CRC Press, Boca Raton, 1998.

5. Carey, G.F., *Computational Grids: Generation, Adaptation, and Solution Strategies*, Taylor & Francis, Washington, D.C., 1997.

6. Shimada, K., Yamada, A., and Itoh, T., "Anisotropic Triangular Meshing of Parametric Surfaces via Close Packing of Ellipsoidal Bubbles," *6th International Meshing Roundtable* (Organized by Sandia National Lab.), 1997.

7. Winslow, A.M. "Equipotential Zoning of Two-dimensional Meshes." Rept. UCRL-7312. University of California, 1963.

8. A. Jameson and D. Mavriplis, 'Finite volume solution of the two-dimensional Euler equations on unstructured triangular mesh', AIAA Paper 87-0435, 1985.

9. D. Mavripilis and A. Jameson, 'Multigrid solution of the two-dimensional Euler equations on a regular triangular mesh', AIAA Paper 87-0353, 1987.

10. J.A. Desideri and A. Dervieux, 'Compressible flow solvers using unstructured grids', Von Karman Institute Lecture Series 1988-05, 7-11 March 1988, pp. 1-115.

11. S.R. Allmaras and M.B. Giles, 'A second order flux split scheme for the unsteady 2-D Euler equations on arbitrary meshes', AIAA Paper 87-1119, 1987.

12. D.J. Mavripilis, 'Accurate multigrid solution of the Euler equations on unstructured and adaptive meshes', AIAA/ASME/SIAM/APS First Natl Fluid Dynamics Congr., 1988.

13. T.J. Barth and D.C. Jespersen,'The design and application of upwind schemes on unstructured meshes', AIAA Paper 89-0366, 1989.

14. S. Sengupta, J Hauser, P.R. Eiseman and J.F. Thompson (eds), Numerical Grid Generation in Computational Fluid Mechanics '88, Pineridge Press, Swansea, 1988.

15. Jones, R.E. "A Self-Organizing Mesh Generation Program." Trans. ASME PVP-13 (1974): 1-7.