ON THE AUTOMATIC GENERATION OF NEAR-OPTIMAL MESHES FOR THREE-DIMENSIONAL LINEAR ELASTIC FINITE ELEMENT ANALYSIS

by

Soo-Won Chae

B.S.M.E., Seoul National University (1977)
M.S.M.E., Korea Advanced Institute of Science and Technology (1979)

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

January 1988

Signature of Author ______________________________
Department of Mechanical Engineering
January 28, 1988

Certified by ______________________________
Professor Klaus-Jürgen Bathe
Thesis Supervisor

Accepted by ______________________________
Professor Ain A. Sonin
Chairman, Graduate Committee

ON THE AUTOMATIC GENERATION OF NEAR-OPTIMAL MESHES FOR THREE-DIMENSIONAL LINEAR ELASTIC FINITE ELEMENT ANALYSIS

by

Soo-Won Chae

Submitted to the Department of Mechanical Engineering on January 28, 1988 in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Abstract

A unified approach to the automatic generation of near-optimal finite element meshes both for two-dimensional and three-dimensional analysis is presented. The procedure is composed of , firstly, an initial mesh construction and, secondly, the *h-version* of adaptive refinement based on an error analysis. For the initial mesh construction in two-dimensional analysis, a robust triangulation scheme is developed, in which triangular elements are generated from the outside boundaries. For three-dimensional applications, a new volume triangulation scheme is developed by employing the same concept, but now tetrahedral elements are generated from the outside surfaces. The adaptive refinement process has been implemented for two-dimensional applications. For this process, an error indicator is introduced with a relaxation factor to obtain efficient solutions. This indicator can be employed in two-dimensional and three-dimensional analysis. The solution processes have been implemented using a data structure with a view toward use of a solid modeler. Examples of mesh generation in two-dimensional and three-dimensional analyses are presented and self-adaptive mesh improvements are given for two-dimensional applications. These example solutions demonstrate that a near-optimal mesh for a given accuracy of solution can be obtained in 2 to 3 steps of iterative refinement.

Thesis Supervisor:	Professor Klaus-Jürgen Bathe
Title:	Professor of Mechanical Engineering

Acknowledgements

I would like to express my deep appreciation to Professor Klaus-Jürgen Bathe for his continuous interest, encouragement and guidance throughout this research and I am very grateful for his support. I am also grateful to Professor David C. Gossard and Professor W. Gilbert Strang for their valuable advices as members of my thesis committee.

I am indebted to the past and current members of our Finite Element Research Group, especially Dr. Theodore Sussman and Mr.Seong-Wook Cho for their comments and suggestions and other friends, Mr.Woo-Chun Choi and Mr.Ho-Myung Chang for their helps in preparing the figures.

I am very grateful to the Korea Science and Engineering Foundation and the Korea Institute of Machinery and Metals for the fellowship which financed my first three years at M.I.T.

I am also grateful to Mr. and Mrs.Jae-shin Lee, the president of Korean Catholic Community of Boston, for their continuous encouragement and support and to Dr. and Mrs. Chongwha Pyun, whose generosity and support have enriched our life in the U.S. and those days of living at their house in Andover have been our favoriate days.

Finally, I want to thank my parents and parents-in-law for their encouragement and my wife, Chung-Hee, for her patience and support. I dedicate this thesis to my daughter, Myoung-Jin, and to all those who in my country have suffered for the good things.

Table of Contents

Chapter 1

Introduction

Due to the increased use of CAD systems, especially finite element analysis, much research effort has been focussed during the last decade on automatizing finite element analysis procedures. For linear analysis, the major problem is the automatic construction of optimal finite element meshes. The concept of an optimal mesh can be described in several ways. We consider the optimal mesh as the one which gives a solution of a given accuracy with the minimum total cost. Since it is not easy, in practice, to construct an optimal mesh accurately, we will consider a near-optimal mesh instead. In our discussion, a near-optimal mesh refers to an efficient mesh that is very close to an optimal mesh.

There are, in general, three different approaches to obtain an optimal mesh. The first approach [1-4], referred to the *r-version*, improves the quality of the finite element solution by considering the nodal coordinates as unknowns in the appropriate energy functional for a fixed mesh topology. Iterative procedures are commonly employed for the required minimization in this approach, but the amount of computational effort is so great that the same accuracy of solution could be obtained with a very fine mesh at less expense. Moreover, if not enough elements are used, the required accuracy cannot be achieved.

The second approach increases the accuracy of the finite element solution by grid enrichment techniques. Here, based on a-posteriori error criteria, the number of mesh degrees of freedom is increased. The two key issues in the application of this approach are the criterion used for mesh refinement and the method of grid enrichment employed. In order to obtain efficient error criteria, much research effort [5-14] has been directed on this subject.

As a grid enrichment method, the *h-version* method is more widely adopted than the *p-version* because of the simplicity involved in the actual implementation. In the *h-version* of refinement [5-9], the total number of elements is increased by reducing the element size where the error measure exceeds a given error tolerance. In the *p-version* of refinement [15-17], the element sizes remain fixed and the degree of polynomial interpolation functions is increased.

Using the grid enrichment techniques, the analyst can obtain a solution within a given accuracy. However, the amount of grid enrichment per cycle is, in practice, limited and the grid enrichment procedure is constrained to the initial mesh configuration. Consequently, the final result of the process may be a less efficient mesh than possible.

The third approach, referred to as the remeshing method, allows for greater refinement in a single cycle than do the above procedures by remeshing the entire domain using the particular solution parameters obtained from an initial analysis, see for example Turke [18-19], Shephard and co-workers [20], and Ladeveze and Leguillon [21].

Turke [18] pointed out that several solution parameters such as isostatics, isoenergetics, isobars of maximum shear stress, and contours of constant displacement can be used as guidelines in constructing near-optimal meshes. Among these parameters, the one of particular interest is the isoenergetics, the contours of constant strain energy density, as suggested by Oliveria [22] and Shephard [23].

In this method, the new mesh in each cycle is not constrained to the previous mesh configuration, but the definition of a new mesh could be more time-consuming than using a selective refinement method, and there is no guarantee that the new mesh yields a solution within a given accuracy.

Therefore, the desirable approach to obtain a near-optimal mesh would be to combine an efficient initial mesh construction and the *h-version* of refinement method. By using a combined method, we can construct a near-optimal mesh in 2 to 3 steps of total solution processes.

We note that the methods discussed above have so far only been applied in two-dimensional analysis. In the case of three-dimensional analysis, only a limited number of algorithms have been developed for the volume triangulation. In these algorithms, only the geometric subdivision of an object has been of concern and the effectiveness of the mesh configuration was not considered.

For the automatic mesh generation, triangular elements are widely adopted in two-dimensional analysis, while tetrahedral elements are employed in three-dimensional analysis, because of their flexibility to fit into any arbitrary analysis domain.

Our goal in this thesis is to construct a near-optimal mesh for a given accuracy of solution in 2 to 3 steps of the total solution process by combining an efficient initial mesh construction and the *h-version* of adaptive refinement method. For this purpose, the automatic mesh generation scheme required for an initial mesh construction, both in two-dimensional and three-dimensional analysis, should be robust and satisfy the following conditions as much as possible:

1. The user should be able to control the local mesh density in any part of the analysis domain.

2. The elements should be as close as possible to equilateral triangles in two-dimensional analysis or equilateral tetrahedra in three-dimensional analysis.
3. The concept of the algorithm should be applicable both for two-dimensional and three-dimensional problems.
4. The algorithm should be economical with respect to both human effort and computer time.

An extensive review of two-dimensional mesh generation schemes has been given by Thacker [24]. The following schemes are available.

1. Methods based on generating internal nodes throughout the domain using a random number generator and construct triangular elements using tesellation operators.[25-28]
2. Methods based on partitioning of the domain into convex subregions and recursively subdividing the subregions down to the element level.[29-30]
3. Spatial discretization methods using modified quadtree encoding techniques.[31-33]
4. Methods generating triangular elements inward from the outside boundary with key nodes on it.[34-36]

Of the four schemes listed above, the 3rd and 4th scheme hold the best possibility of satisfying the above conditions for being a good mesh generation scheme. However, in the 3rd scheme, the control of local mesh density is not as flexible as for the 4th scheme, while in the 4th scheme, ill-conditioned elements are occasionally generated and the extension to three-dimensional analysis is not available.

In the case of three-dimensional mesh generation, only a limited number of algorithms have been developed. As summarized by Shephard [37], the approaches used for automatic mesh generation for three-dimensional solids include:

1. Volume triangulation of a set of points placed on the surface and inside the object.[38-39]
2. Recursive subdivision of the object down to the element level.[40]
3. Spatial enumeration by employing modified octree encoding techniques.[41-43]
4. Paring topologically simple volumes from the object one at a time.[44-45]

The first three approaches employ the same concepts of the ones in two-dimensional case. The 4th approach is similar to the 4th one in the two-dimensional case because it starts from the outside boundary , but it differs because the user cannot assign the local mesh density in this case.

In the first approach, the placement of nodal points on the surface and inside the object is a very complicated task, and thus it is not suitable for general three-dimensional applications in practice. The 2nd and 4th approaches are not appropriate for graded mesh generations and the resulting meshes can be ill-conditioned. Among the above approaches, the 3rd approach has shown, so far, the best possibility of satisfying the conditions mentioned above. However, it would be desirable if an algorithm, which employs the concept of the 4th approach in two-dimensional analysis, were developed for three-dimensional analysis, in order to satisfy the above conditions as much as possible.

Our contribution in this thesis is to develop a synthesized mesh generator which constructs near-optimal meshes for a given accuracy of solution both in two-dimensional and three-dimensional linear elastic finite element analysis.

Specifically, the entire process from constructing the initial mesh to the adaptive refinement process has been developed for two-dimensional analysis. For three-dimensional analysis, the initial mesh construction and the error analysis have been developed. To obtain the initial mesh, a robust triangulation scheme employing the 4th approach is developed for two-dimensional analysis and a new volume triangulation scheme is developed by employing the same concept.

As for the adaptive refinement process, an error indicator is proposed, which is a variation of theoretical error indicators suggested by Babuska and et al..[5-9] The indicator is based on our numerical observations to obtain the maximum accuracy of solution during the refinement process. The *h-version* of grid enrichment scheme is employed for the refinement process, in which each element is subdivided into four subelements by halving the element size.

In order to implement the initial mesh construction from a solid modeler and the refinement process, a data structure called *dissembled winged-edge/face data structure* is designed, which employs the concepts of the winged-edge data structure commonly used in CAD/CAM systems [46-48].

Throughout the thesis, only quadratic elements are considered because of their effectiveness in the analysis. They are 6-node triangular elements and 8-node quadrilateral elements and 10-node tetrahedral elements.

In the following chapters, we present the algorithms we have developed for near-optimal mesh generation and example solutions.

In chapter 2, we consider the two-dimensional problem, which includes the initial mesh construction and the adaptive refinement process for 6-node triangular elements and 8-node quadrilateral elements.

In chapter 3, three-dimensional problems are considered, which includes the initial mesh construction and the error analysis. The adaptive refinement process for three-dimensional problems is not included in the thesis.

In chapter 4, we present the example solutions which show the effective use of our algorithms developed for near-optimal mesh constructions.

Finally, Chapter 5 presents the conclusions of the thesis and recommendations for future work.

Chapter 2

Mesh Generation for Two-Dimensional Problems

2.1 Overall Procedures for Near-Optimal Mesh Generation

The schematic diagram for our near-optimal mesh generation for two-dimensional problems is shown in Figure 2-1. As shown in Figure 2-1, the near-optimal mesh construction schemes for 6-node triangular elements and 8-node quadrilateral elements are different at the initial mesh construction stage. For 6-node triangular elements, the whole process from the construction of the initial mesh using the solid modeler to the adaptive refinement process based on our error analysis can be handled automatically. However, in the case of 8-node quadrilateral elements, the initial mesh should be input manually and only the adaptive refinement process is performed automatically. The adaptive refinement process in Figure 2-1 can be considered to equalize the error indicators throughout the analysis domain by refining the element where the error indicator exceeds the given error tolerance.

All the above procedures are implemented in a computer program AMESH, which is composed of AMESH-I and AMESH-II. AMESH-I is designed to construct the initial mesh using the solid modeler input for triangular elements, and AMESH-II is designed to perform the adaptive refinement process both for 6-node and 8-node elements.

The solid modeler used in AMESH-I is of boundary representation type and it will be necessary to use an interfacing program to prepare the input data for AMESH-I if an external solid modeler (either CSG or B-REP type) is to be linked to AMESH-I directly.

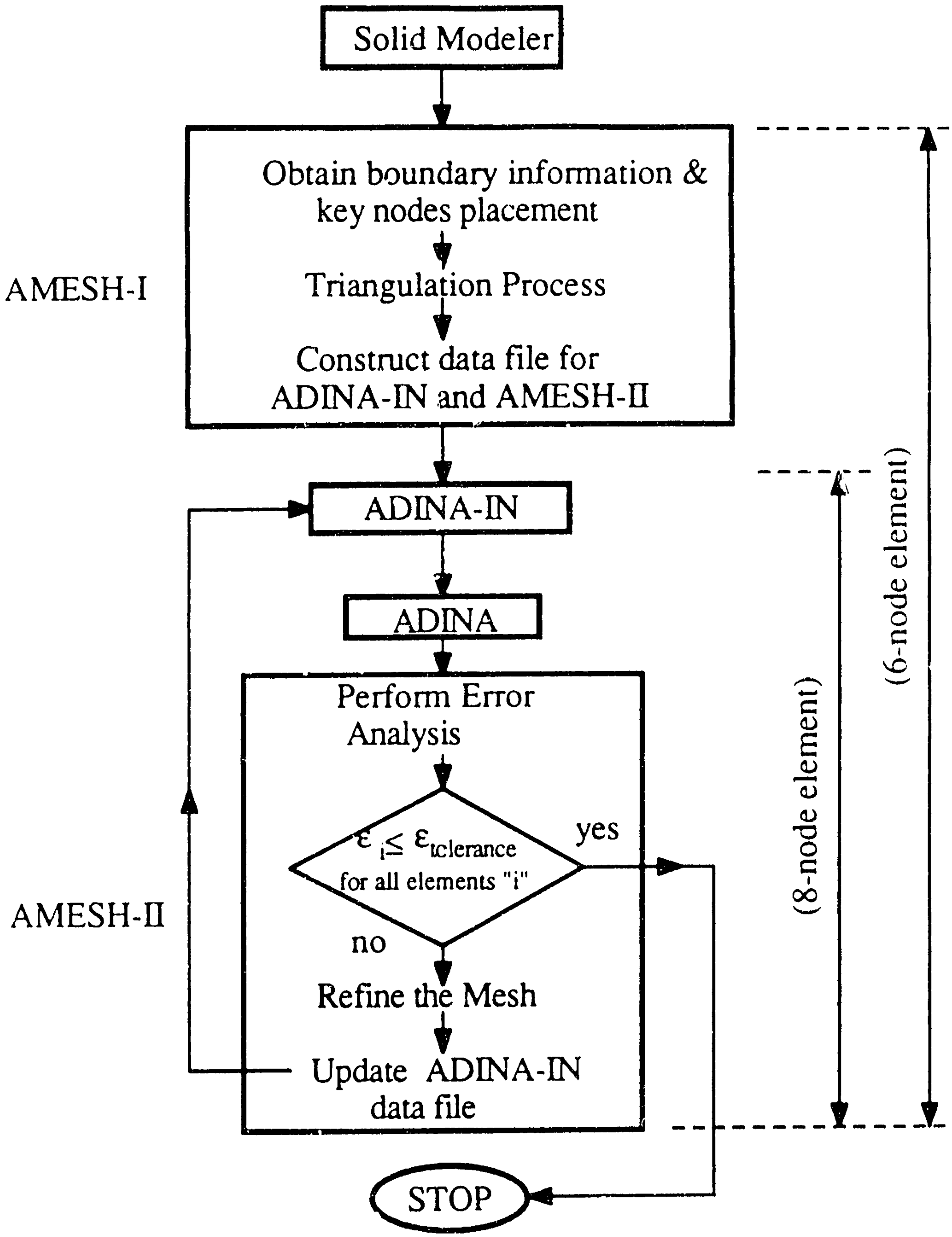

Solid Modeler
Obtain boundary information & key nodes placement
Triangulation Process
Construct data file for ADINA-IN and AMESH-II
AMESH-I
ADINA-IN
ADINA
Perform Error Analysis
$\varepsilon_i \leq \varepsilon_{tolerance}$ for all elements "i"
yes
no
Refine the Mesh
Update ADINA-IN data file
AMESH-II
STOP
(8-node element)
(6-node element)

Figure 2-1: Near-optimal mesh generation process for 2-D problems

The solid modeler in AMESH-I is designed to store the minimum amount of data required for mesh generation and only to include the system with straight lines and circular arcs. Hence, there is no redundant information stored and the data to be stored can be easily input by the user or extracted from the external solid modeler if used.

As already pointed out in Chapter.1, since the meshes that are constructed during the adaptive refinement process are restraint to be "contained" in the initial mesh and the grid enrichment is limited to 4 times per each original element, a start-up solution (i.e., starting with a new mesh) may be required to obtain the given accuracy of solution if an initial mesh is very coarse. Compared with the adaptive refinement process in which ADINA-IN [49], ADINA [50] and AMESH-II are used in every iteration, the total cost required for AMESH-I is much less, because only geometric considerations are involved. Hence, sometimes it is more efficient to reconstruct the initial mesh rather than to perform the adaptive refinement process several times, even while the refinement process has already been started.

Therefore, the primary goal of this mesh generation process is to construct the initial mesh as close as possible to an optimal mesh for the required accuracy of solution. Here the users' experience and other general guidelines need be used in order not to perform the adaptive refinement process more than 2 to 3 times.

2.2 Mesh Generation and Adaptive Refinement Process for 6-Node Triangular Elements

2.2.1 Triangulation Process

The proposed scheme for automatic triangulation is a modified version of Sadek's algorithm [35]. In this section, the detailed description of the modified Sadek algorithm is given, pointing out what modifications have been made to the original algorithm and why such modifications have been made. The basic strategy of the method is as follows : To begin, key nodes are placed around the boundary considering the desired local mesh density and ordered in the counter-clockwise direction to form a loop-boundary so that the unmeshed region lies to the left as we travel along a loop-boundary. Given a certain domain with a number of nodes around its boundary, triangular elements are generated from the outside boundary toward inside by cutting the corner nodes of the boundary as shown in Figure 2-2. If an analysis domain has a complex shape, it is recommended, in our scheme, that the entire domain be subdivided into near convex subdomains, in order to construct well-conditioned triangular elements. The reason for this restriction is explained later in this chapter.

Since the best form of triangular elements in finite element analysis is known to be the equilateral element, the resulting mesh will be generated as close as possible to have equilateral triangles. Hence, the number of triangular elements that can be generated at a node "i" with a boundary angle ϕ_i is taken as the nearest integer of $\phi_i/\pi/3$ so that the triangle is as equilateral as possible.

In Sadek's algorithm, a corner node is defined as the one at which the boundary angle is not equal to 180°. According to this definition, a corner node includes the case when the boundary angle is greater than 180° as well as the case

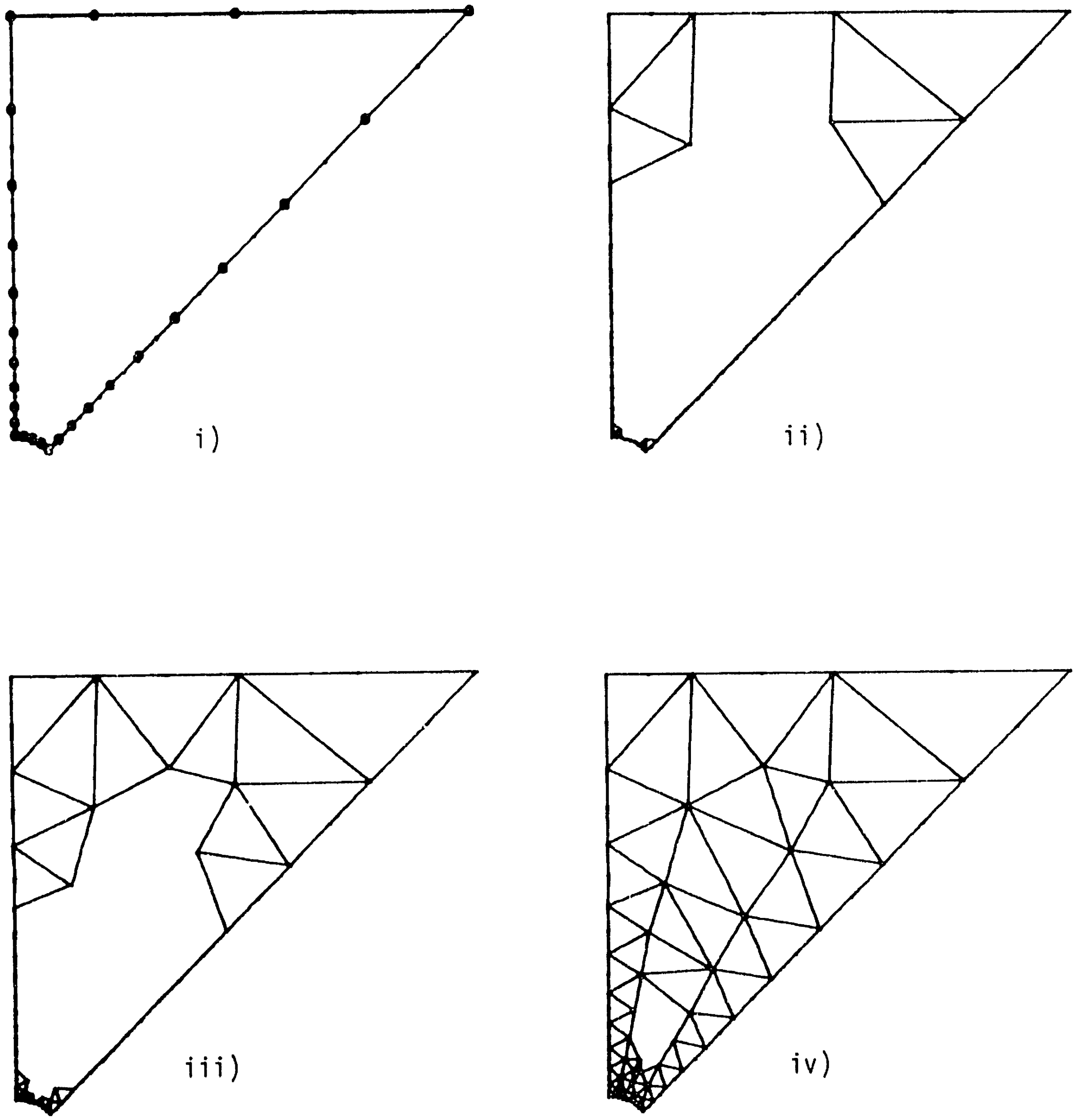


Figure 2-2: Example of triangulation process

less than 180°, which implies that the basic operations that construct the elements at a corner node should be designed to generate from one element up to six elements. But the details of these operations are not shown in his work.

At the initial stage of mesh generation when many of the boundary angles are still 180°, the above corner node may make sense, but as the triangulation process proceeds, the resulting boundary angles are not usually 180°, so almost all of the loop-boundary nodes are corner nodes. In such a case, there is no reason to continue with the 180° as a criterion for a corner node.

Our triangulation scheme can be considered to generate elements by *trimming* or *digging* a key node in a loop-boundary to reduce the number of key nodes and if the number of key nodes reaches three, the triangulation process is finished. In that sense, in order to construct well-conditioned elements, at least two basic operations are needed. One is to generate one element by *trimming* a well-conditioned key node and the other is to generate two elements by *digging* into a loop-boundary at a key node, which is to promote producing well-conditioned key nodes.

Therefore, two basic operations (Type-1 and Type-2 operation) are designed in our scheme and the corner nodes for these operations that can generate one or two elements at a key node, can have a maximum boundary angle of 150°. The corner nodes for these operations are type-1 and type-2 corner node respectively. In addition, one more operation (Type-0) is designed to complete the triangulation process, which constructs the last two elements when the number of key nodes reaches four.

The Type-1 operation, as shown in Figure 2-3, is designed to generate one element by trimming one type-1 key node which satisfies certain conditions that are described later in this chapter, in order to construct well-conditioned elements.

In Sadek's algorithm, the user is supposed to input the criteria for type-1 node decision and the details are not explained. However, to our experience, this criterion is one of the key issues in the triangulation process. Many of the existing triangulation schemes [34][36] use the boundary angle as a criterion for type-1 node decision such as $\phi_i \leq 85°$ or $\phi_i \leq 90°$. But our experience shows that in addition to the boundary angle, the size of adjacent edges and the effects of the Type-1 operation on the loop-boundary should all be considered as criteria for type-1 node decision. The resulting heuristic criteria are described later in this chapter.

The Type-2 operation is designed to generate two triangular elements at a type-2 corner node by introducing a new key node as shown in Figure 2-4. In our research, an *adaptive* Type-2 operation has been designed, in which the position of a new generated key node is adjusted according to the scaling factor, r, in order to avoid an overlapped region or bottle-neck like region in a loop-boundary. The implementation of this *adaptive* operation is also a modification to the original algorithm. The *adaptive* Type-2 operation is composed of two steps. First, a new key node is generated considering the effects of neighbouring nodes using the method developed by Sadek as shown in Figure 2-5. As a second step, the new key node position is adjusted according to the scaling factor, r which is described later in this chapter. Consider the first step of a Type-2 operation at node 3 in Figure 2-5 b). In order to construct two well-conditioned elements, the ratio l_2/l_{23} is set to be equal to the ratio l_{23}/l_3, i.e., the length l_{23} is taken as $\sqrt{l_2 l_3}$. The position A_1 is determined by the length l_{23} taken along a line bisecting the angle ϕ_3.

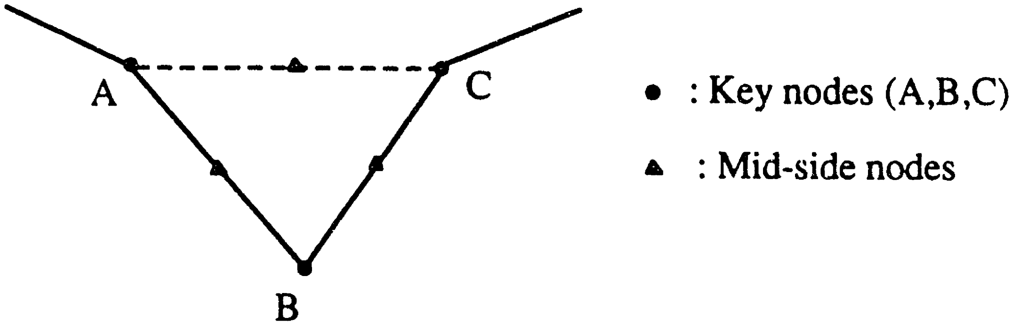


Figure 2-3: Example of Type-1 operation

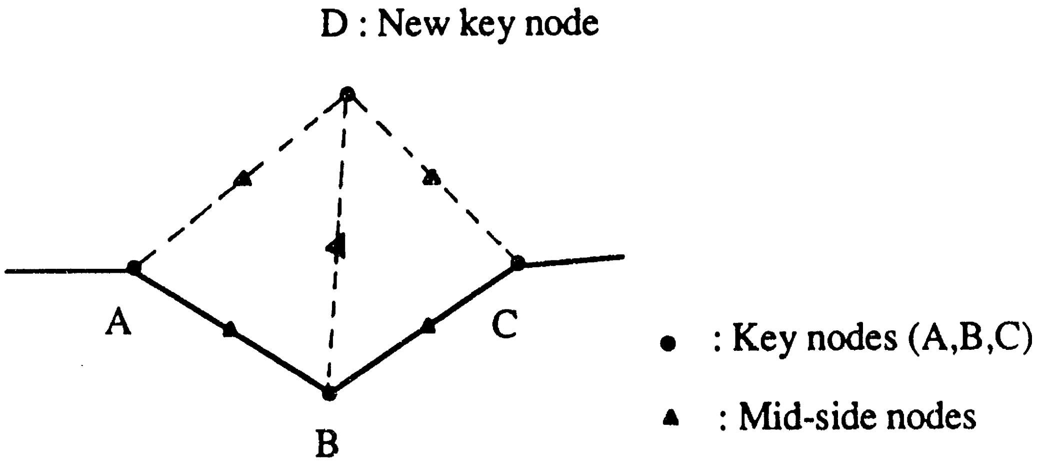


Figure 2-4: Example of Type-2 operation

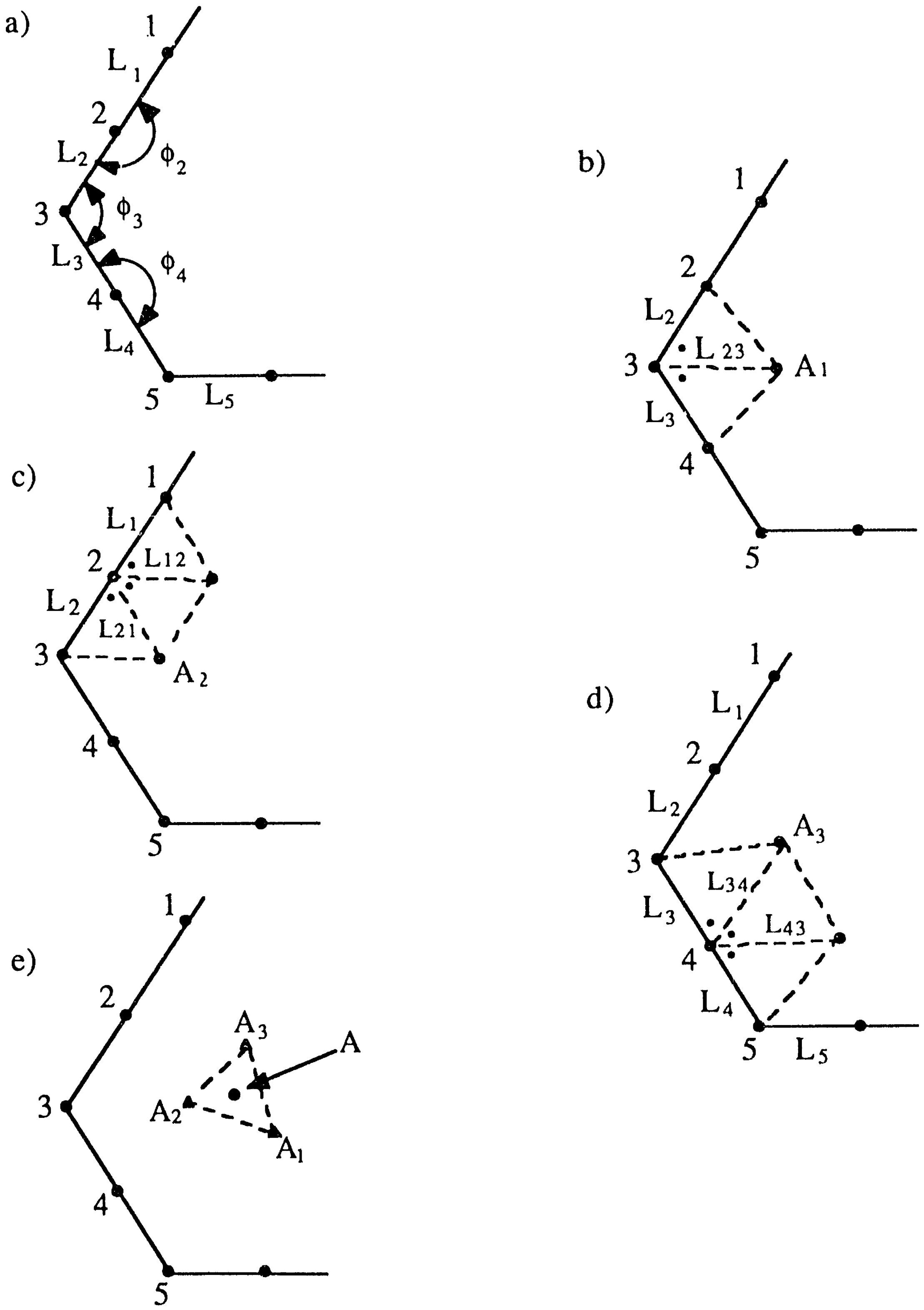


Figure 2-5: Generation of a new key node [35]

$$\frac{l_2}{l_{23}} = \frac{l_{23}}{l_3}, \quad or \quad l_{23} = \sqrt{l_2 l_3} \qquad (2\text{–}1)$$

In order to include the effects from neighbouring nodes such as node 2 and node 4, the same procedure is repeated at these nodes. But if the angle at he neighbouring node is greater than 210° when more than three elements should be generated, its effect is not included for brevity. The number of triangular elements that can be generated at a node "i" with a boundary angle ϕ_i is taken as the nearest integer of $\phi_i/\pi/3$ so that the triangle is as equilateral as possibie. Assume that angle ϕ_i at node 2 is 180° and thus three triangles can be constructed. The condition that makes the three triangles well-conditioned is

$$l_1/l_{12} = l_{12}/l_{21} = l_{21}/l_2$$

$$or \qquad (2\text{–}2)$$

$$l_{12} = {}^3\sqrt{l_1{}^2 \cdot l_2} \quad and \quad l_{21} = {}^3\sqrt{l_1 \cdot l_2{}^2}$$

Similarly, l_{2l} and $\phi_2/3$ determines the position A_2. Another new node A_3 can be generated at node 4. In order to choose a unique position, A, for a new node from these three positions, weighting factors are used. Since the shape of small elements will be more distorted by using a modified new node, the weighting factors are taken in the inverse proportion to the size of the resulting elements, i.e.

$$w'_1 \propto \frac{1}{l_{23}^2}\,, \quad w'_2 \propto \frac{1}{l_{21}^2}\,, \quad \textit{and} \quad w_3 \propto \frac{1}{l_{34}^2}$$

and (2–3)

$$w'_1 + w'_2 + w'_3 = 1$$

The actual values of w'_1, w'_2, w_3 are taken as

$$w'_1 = \frac{1}{l_{23}^2}\cdot\frac{1}{w'_0}\,, \quad w'_2 = \frac{1}{l_{21}^2\cdot w'_0}\,, \quad \textit{and} \quad w_3 = \frac{1}{l_{34}^2\cdot w_0}$$

$$\textit{where} \quad w'_0 = \frac{1}{l_{23}^2} + \frac{1}{l_{21}^2} + \frac{1}{l_{34}^2} \qquad (2\text{–}4)$$

Therefore,

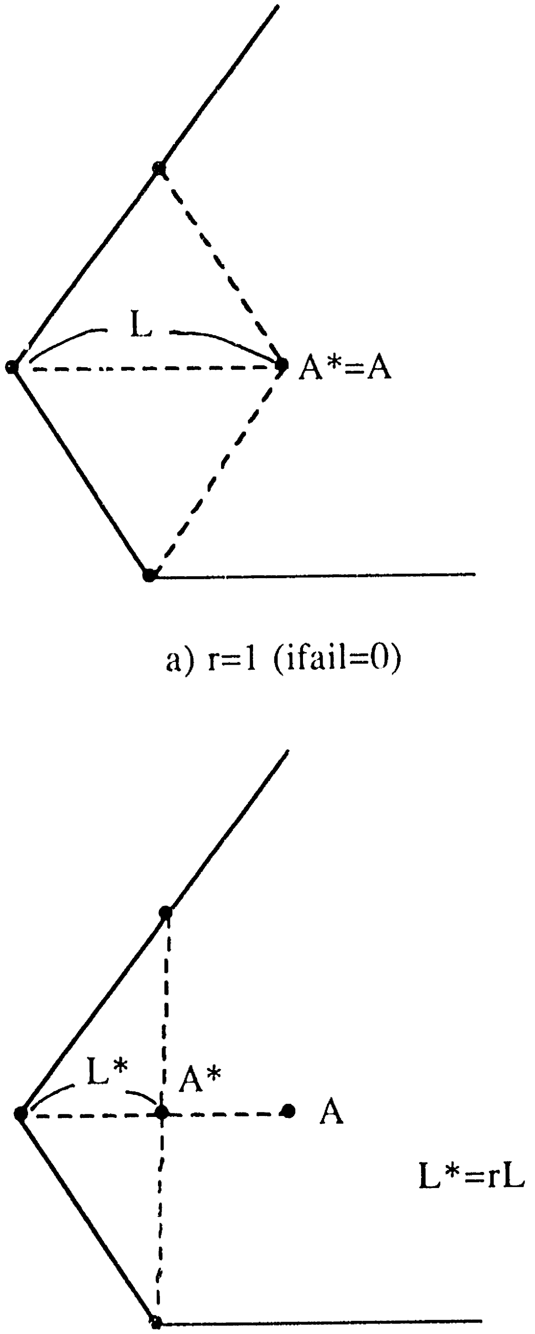
$$A = w'_1A_1 + w'_2A_2 + w'_3A_3 \qquad (2\text{–}5)$$

The next step of an *adaptive* Type-2 operation is to adjust the key node position from A to A* using a scaling factor, *r* as shown in Figure 2-6. The scaling factor, *r*, is defined as

$$r = \frac{1}{1 + \textit{ifail}}\,, \qquad \textit{ifail} = 0, 1, 2, \ldots \qquad (2\text{–}6)$$

where *ifail* is a number that depends upon whether a basic operation could not be performed. Initially, *ifail* is set to zero and if no more operation is possible because of the failure in check processing, *ifail* is increased by one and so on.

The Type-0 operation is designed to construct the last two elements



a) r=1 (ifail=0)

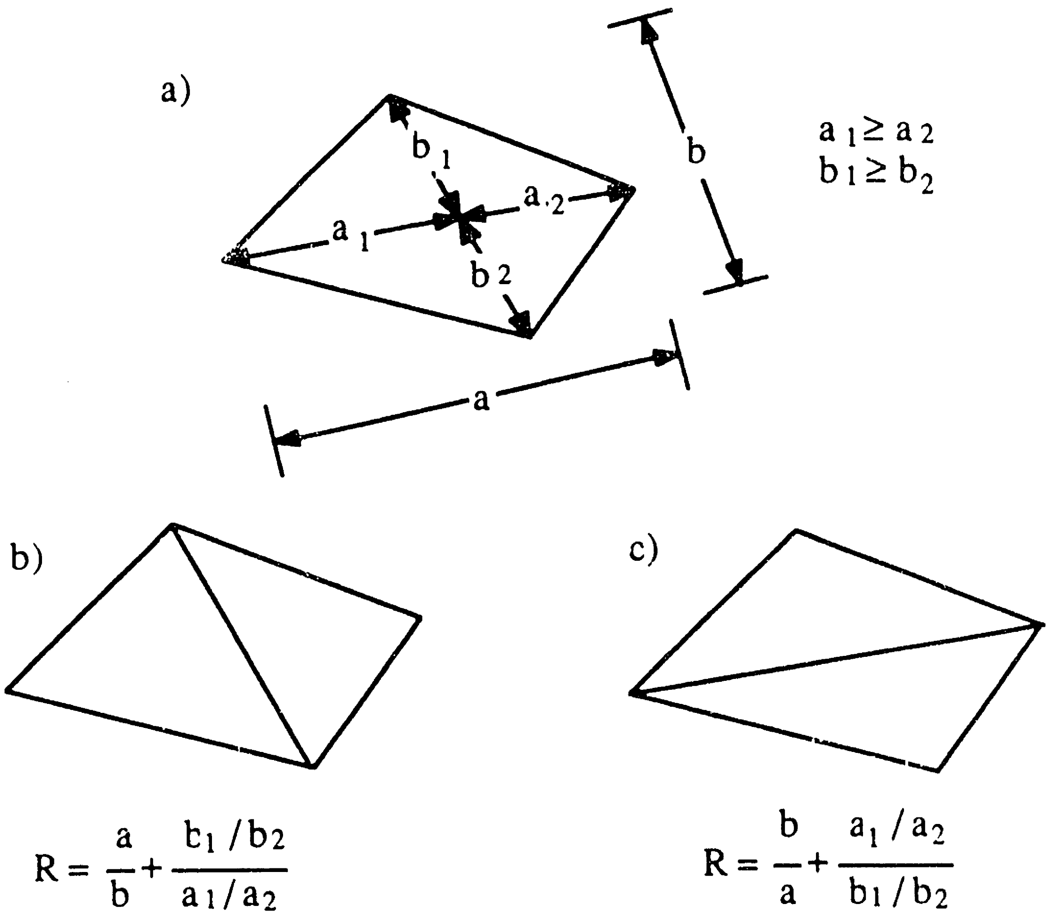
b) r=0.5 (ifail=1)

Figure 2-6: Adjustment of a new key node

when the number of key nodes on the boundary reaches 4 and the remaining area is quadrilateral. It can be divided into two triangles by a diagonal. Zienkiewicz [51] has suggested that a quadrilateral should be divided into two triangles according to the shorter diagonal for better shape conditioning. But Sadek has shown that this scheme does not work well in some cases and suggested another method, in which the subdivision is performed using a factor R as shown in Figure 2-7. Sadek's scheme works well. However, we can obtain a similar result more efficiently by using the boundary angles, which are already available, without calculating the lengths of partitioned diagonals. The one we suggest is that the division is made according to the diagonal for which the sum of diagonal angles is larger, see Figure 2-8.

The nodes on the original boundary are considered to be of level equal to one and the new generated nodes from the original boundary nodes are assigned to level two and so on. In Sadek's algorithm, the level concept has been used to indicate a certain hierarchical order such that a boundary node of level two can be cut to form elements only after all the boundary nodes of level one have been removed.

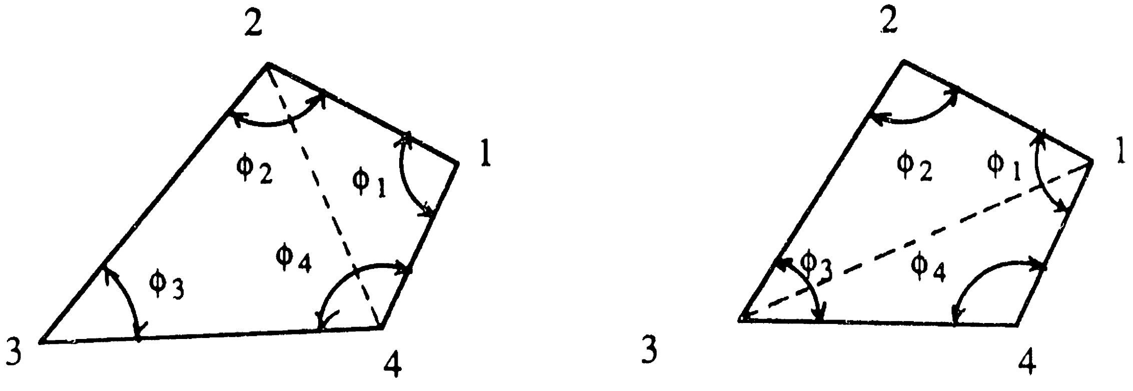
This works well when the aspect ratio of an analysis domain is close to one. But if the aspect ratio is much larger than one at the beginning, it gets larger due to the removal of the layer from the boundary, which may result in a bottle-neck like region in a loop-boundary. Hence, as a modification in our scheme, the level of a node has been used only to determine the order of the basic operations among the candidate nodes on the current loop-boundary nodes. Therefore, conceptually, our scheme is to generate elements by cutting off the sharp corners ($\phi_i \leq 150°$) from the boundary, while Sadek's algorithm is considered to generate elements by removing a boundary layer from the boundary.



$$R = \frac{a}{b} + \frac{b_1/b_2}{a_1/a_2}$$

$$R = \frac{b}{a} + \frac{a_1/a_2}{b_1/b_2}$$

Figure 2-7: Subdivision of a quadrilateral by Sadek[35]



a) $\phi_2 + \phi_4 > \phi_1 + \phi_3$

b) $\phi_1 + \phi_3 > \phi_2 + \phi_4$

Figure 2-8: Suggested Type-0 operation

Since only the sharp corners are cut off from a boundary, the concave boundary nodes can not be removed unless they become convex sharp corner nodes. Hence, we impose a restriction that near convex subdomains are recommended in dividing the complex analysis domain to construct well-conditioned elements. This is especially important when the concave region in the analysis domain has high local mesh density than other regions.

If the element size changes drastically along a loop-boundary, a loop-boundary may sometimes overlap itself or have a bottle-neck like region, thus generating an ill-conditioned mesh , or even make further operations impossible. In order to avoid this unstable phenomenon, the following check processing is designed in our scheme. We call the checking an *overlap check* and *minimum distance check.*

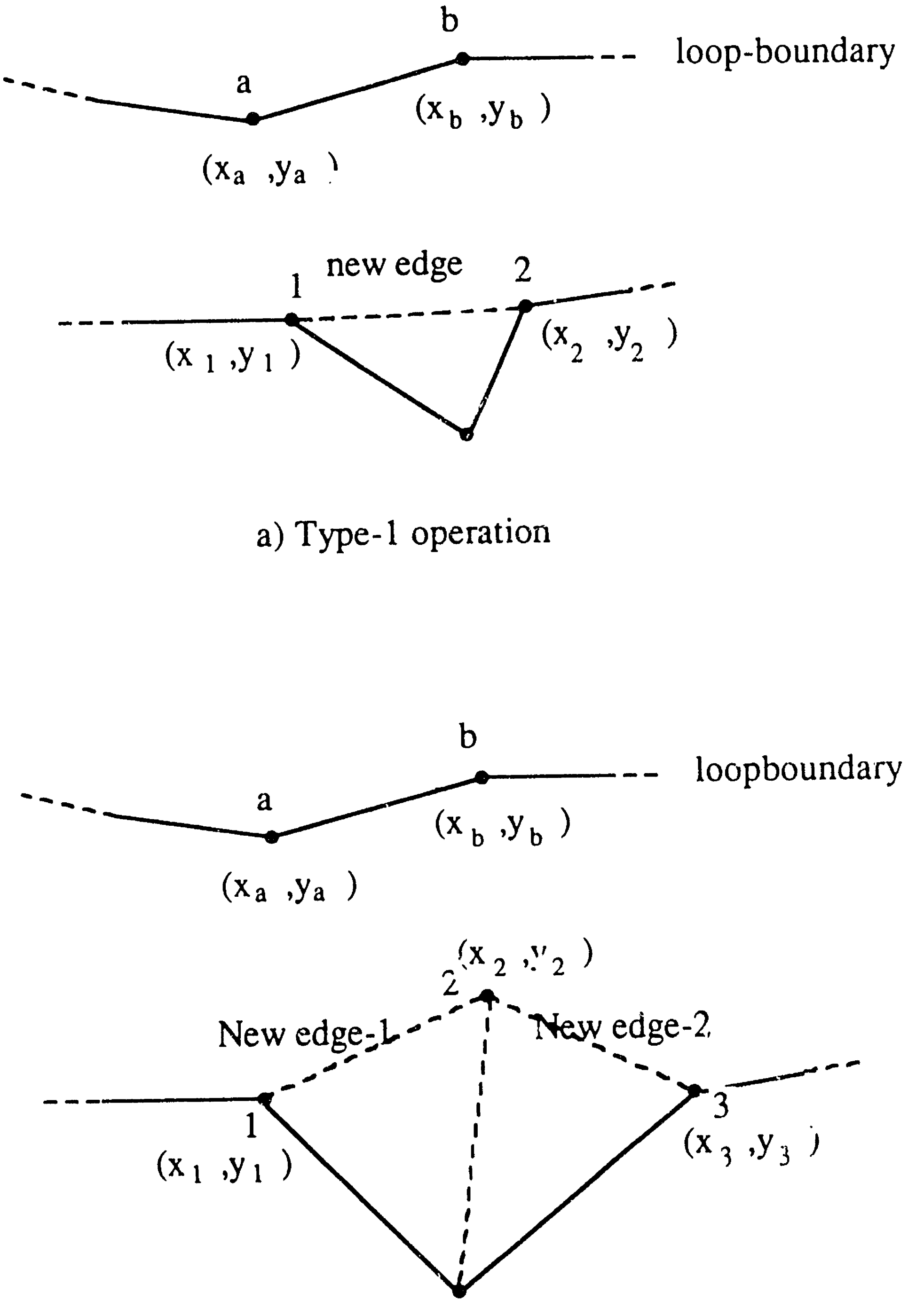
In the *overlap check,* the new edges of the generated triangular elements are checked whether any overlaping occurs between these edges and the remaining loop-boundary edges. As shown in Figure 2-9, in the Type-1 operation, one new edge is to be tested, while in the Type-2 operation two edges are to be tested.

Consider a new edge of Type-1 operation in Figure 2-9(a). The equation of a line which passes through the endpoints, node 1 and node 2, of this edge is found to be

$$y_{21} \cdot x + x_{12} \cdot y + y_1 x_2 - y_2 x_1 = 0 \qquad (2\text{–}7)$$

$$\textit{where} \qquad y_{21} = y_2 - y_1 \ , \quad \textit{etc.}$$

A loop-boundary edge with the endpoints *a* and *b*, may be expressed in parametric form as



a) Type-1 operation

b) Type-2 operation

Figure 2-9: Overlap check processings

$$x = x_a + t \cdot x_{ba}$$

$$y = y_a + t \cdot y_{ba} \qquad (2-8)$$

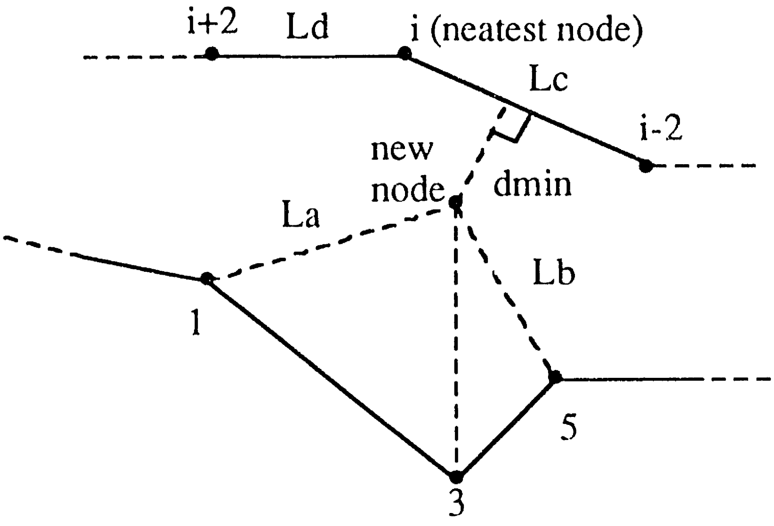
$$where \qquad 0 \le t \le 1$$

The intersection points of a new edge and a loop-boundary edge can be computed by substituting equation (2-8) into equation (2-7). If t lies in the range zero to one, an intersection point should again be tested whether it lies between node 1 and node 2. All the loop-boundary edges should be tested for overlaping with a new edge. In the Type-2 operation, the same tests are performed for two new edges , see Figure 2-9(b).

In the *minimum distance check*, a new key node of Type-2 operation is tested whether the new node has enough distance to the remaining loop-boundary edges in order to avoid any bottle-neck like region in the loop-boundary. The details are shown in Figure 2-10.

The adaptive coefficient, α, used in Figure 2-10, gets smaller to reduce the minimum distance requirement as the triangulation process proceeds and no more Type-1 and Type-2 operations are possible because the overlap check or minimum distance check is not passed.

Because of the successive steps from the boundary toward the inside of the domain, the triangulation process is a path-dependent one. It has been observed that the critical issues in this process are how to define type-1 and type-2 corner nodes among the loop-boundary nodes and in which order the basic operations should be performed. Thus the proposed method in this research has



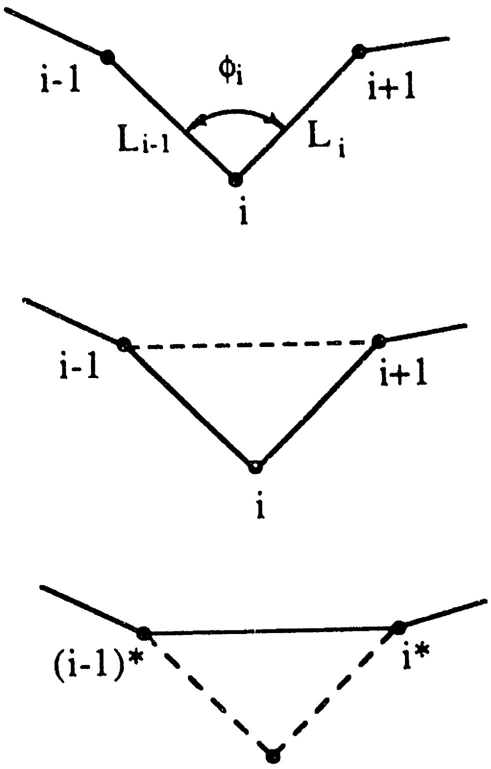
$$\text{dmin} \geq \alpha \quad \text{Average (La, Lb, Lc, Ld)}$$

$$\text{where} \quad \alpha = \frac{0.5}{1 + \text{ifail}} \quad , \qquad \text{ifail}= 0, 1, 2, ...$$

Figure 2-10: Minimum distance check

some heuristic rules which have been identified in an effort to imitate human like reasoning during the triangulation process. The following rules are used to decide whether a boundary node is a type-1 or type-2 node and the order of basic operations. As already mentioned, this is also a modification adopted in our scheme.

1. A type-1 corner node as shown in Figure 2-11 is a loop-boundary node at which the boundary angle, ϕ_i is less than 80° ($\delta_i \geq 1.8$) or 95° ($\delta_i < 1.8$) depending on the edge length ratio, δ_i, and the neighbouring edge length ratios around node i should not be changed drastically after a Type-1 operation. The computer implementation of a type-1 node decision is a little more complicated, however Figure 2-11 shows the basic idea for a type-1 corner node decision.
2. A type-2 corner node is a node for which the boundary angle is less than or equal to 150°, i.e. suitable for two well-conditioned elements and is not a type-1 corner node.
3. The Type-1 operation should be performed before the Type-2 operation. Among the same types of operations, the order of operation is also an important factor especially for Type-2 operations.
4. Type-1 nodes are sorted to decide the order of operations considering the following factors successively.
    - Low level
    - Small boundary angle, ϕ_i
    - Large edge length ratio, δ_i
    - Small adjacent edge length, l_i
5. Similarly, type-2 nodes are sorted to decide the order of operations considering the following factors.
    - Low level
    - Large edge length ratio, δ_i
    - Small adjacent edge length, l_i
    - Small boundary angle, ϕ_i



$$\delta_i = \max\left(\frac{L_{i-1}}{L_i}, \frac{L_i}{L_{i-1}} \right) \geq 1$$

; edge length ratio

before Type-1 operation

$$\delta_{max} = \max(\delta_{i-1}, \delta_i, \delta_{i+1})$$

after Type-1 operation

$$\delta^*_{max} = \max(\delta^*_{i-1}, \delta^*_i)$$

```
If $\phi_i \leq 60°$ then type-1 node

Elseif $60° < \phi_i \leq 80° \sim 95°$ then

        if $\delta_{max} \leq 1.2$ then
                if $\delta_{max}^* \leq 1.5\,\delta_{max}$ then type-1 node
        elseif $1.2 < \delta_{max} \leq 1.5$ then
                if $\delta_{max}^* \leq 1.3\,\delta_{max}$ then type-1 node
        else
                if $\delta_{max}^* \leq 1.2\,\delta_{max}$ then type-1 node
        endif

Else

Endif
```

Figure 2-11: Example of Type-1 corner node decision

After the triangulation process has been completed for a given subregion domain using the above-mentioned basic operations, the resulting mesh is finally improved by the application of a smoothing technique. The smoothing technique adopted in this research is the one suggested by Shephard [23]. In this technique, interior nodes are placed at the average of the centroid of neighbouring nodes and the current location. During this process, the boundary nodes that have been placed by the user remain unchanged. For example,

$$X_{new} = \frac{1}{2}(X_{centroid} + X_{old}) \qquad (2\text{-}9)$$

$$where \qquad X_{centroid} = \frac{1}{N}\sum_{i}^{N} X_i \,|_{nodes\ connected}$$

$$if \quad |X_{new} - X_{old}| \le e \quad for\ all\ the\ internal\ nodes,$$

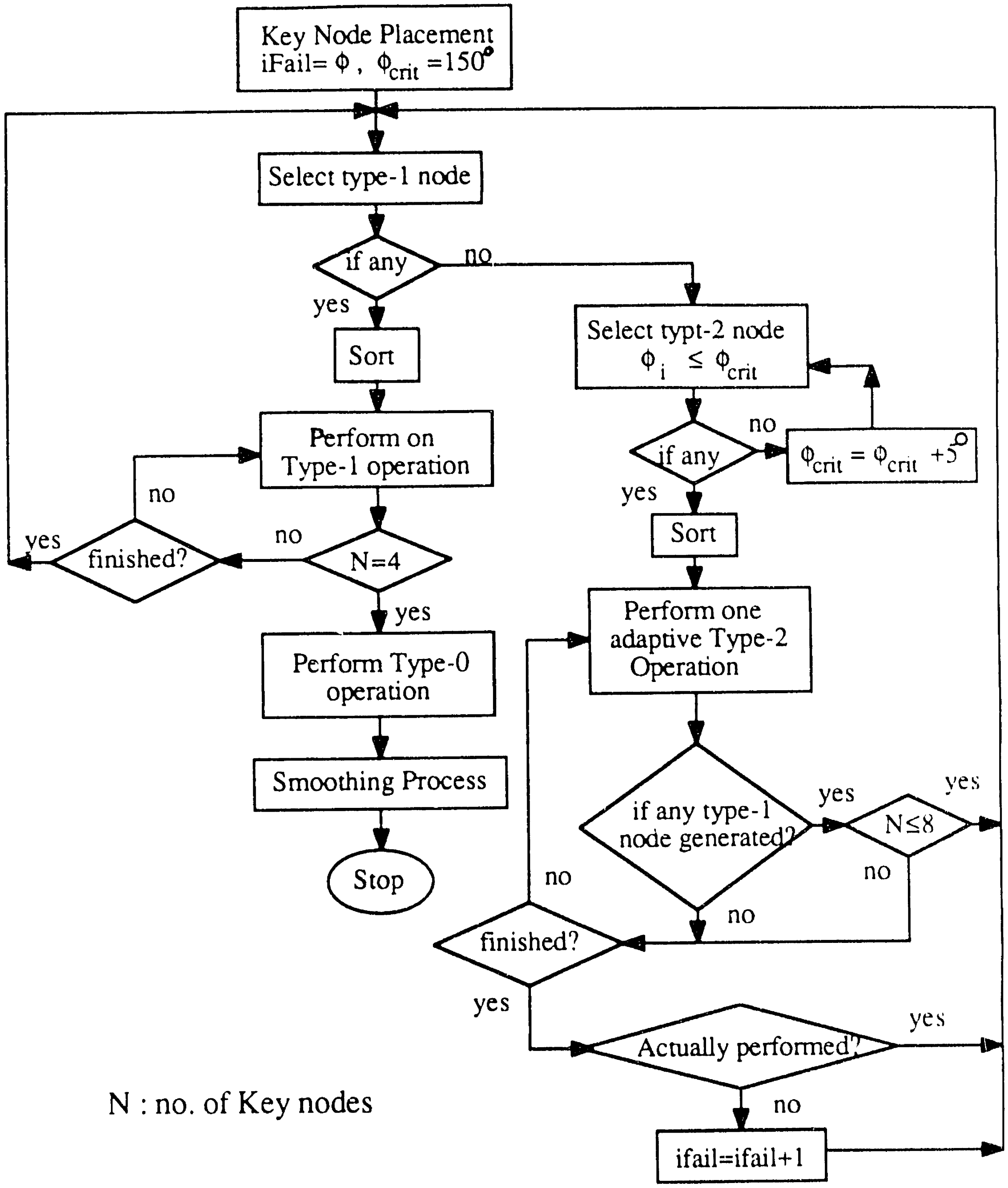
the smoothing process has converged.

$$where \qquad e = L_{system}/500$$

It has been observed that usually 3-6 iterations in this smoothing process are enough to converge to a tolerence of $L_{system}/500$.

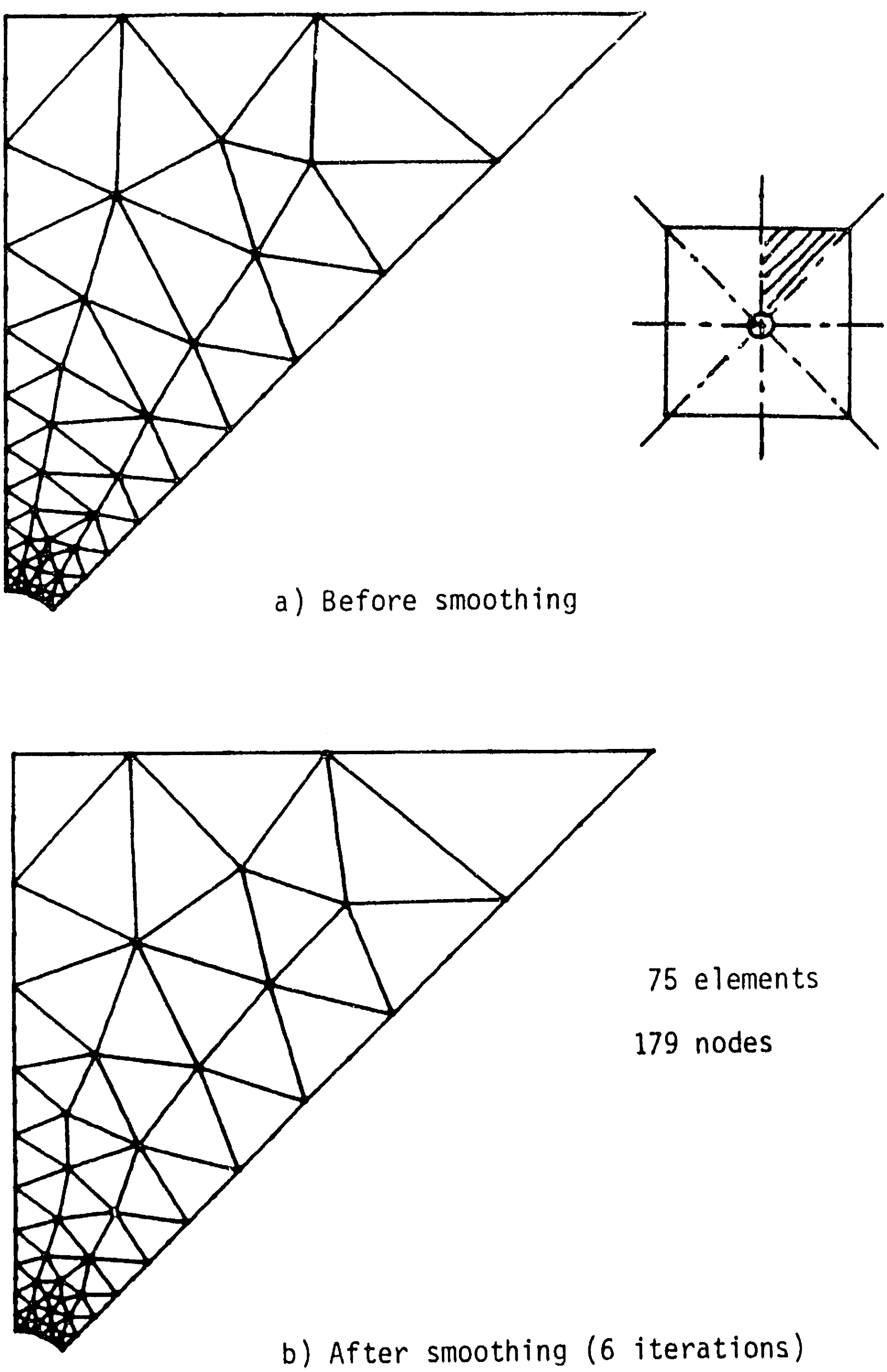
All the above details are summarized in the flow chart of the entire triangulation procedures as shown in Figure 2-12.

Some examples of the triangulation process developed in this research are shown in Figure 2-13 to Figure 2-16. Figure 2-13 shows that the mesh can be improved by using the smoothing process. In Figure 2-14 and 2-15 , complex domains are divided into near convex subdomains in order to construct well-conditioned elements.



N : no. of Key nodes

Figure 2-12: Flow chart of triangulation process



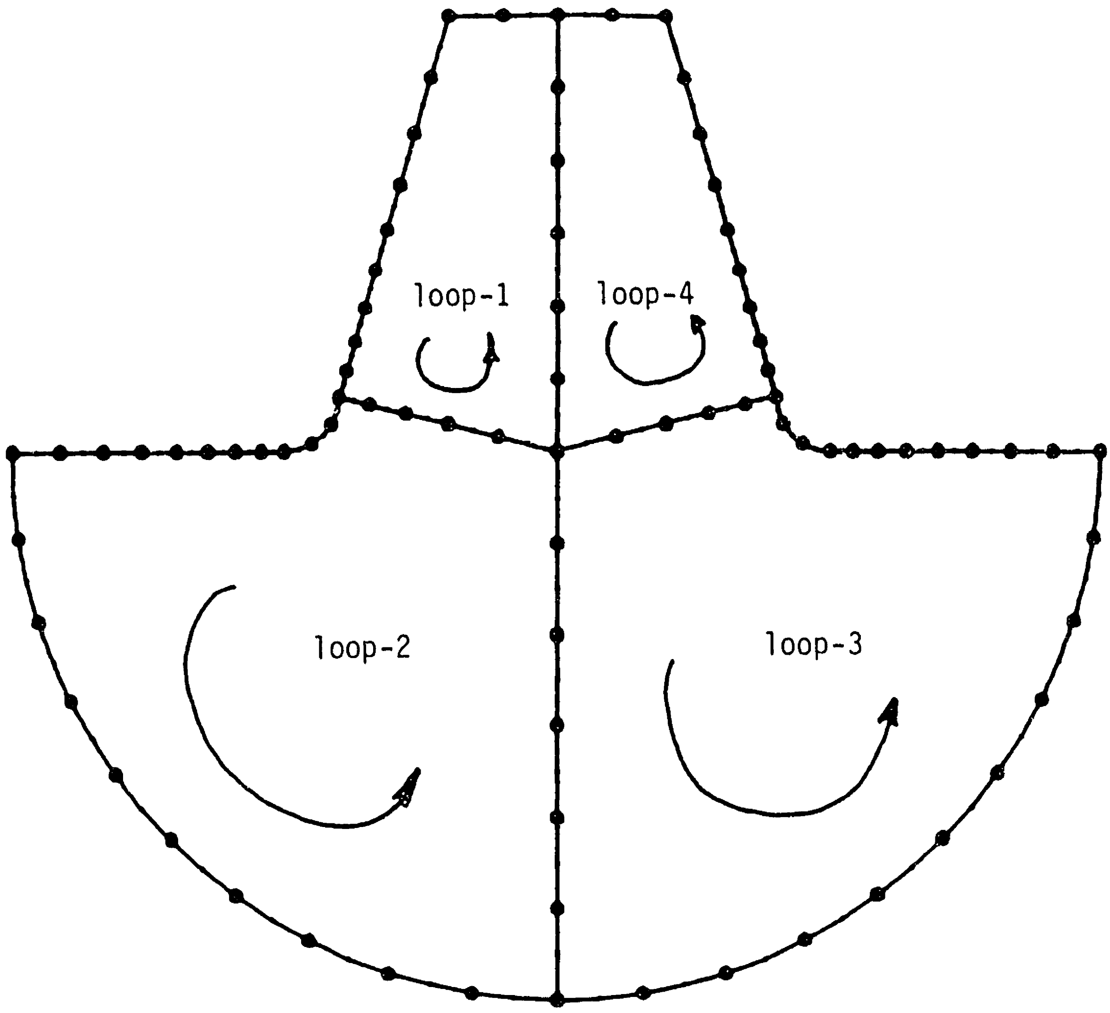
a) Before smoothing

75 elements

179 nodes

b) After smoothing (6 iterations)

Figure 2-13: Square plate with a central hole

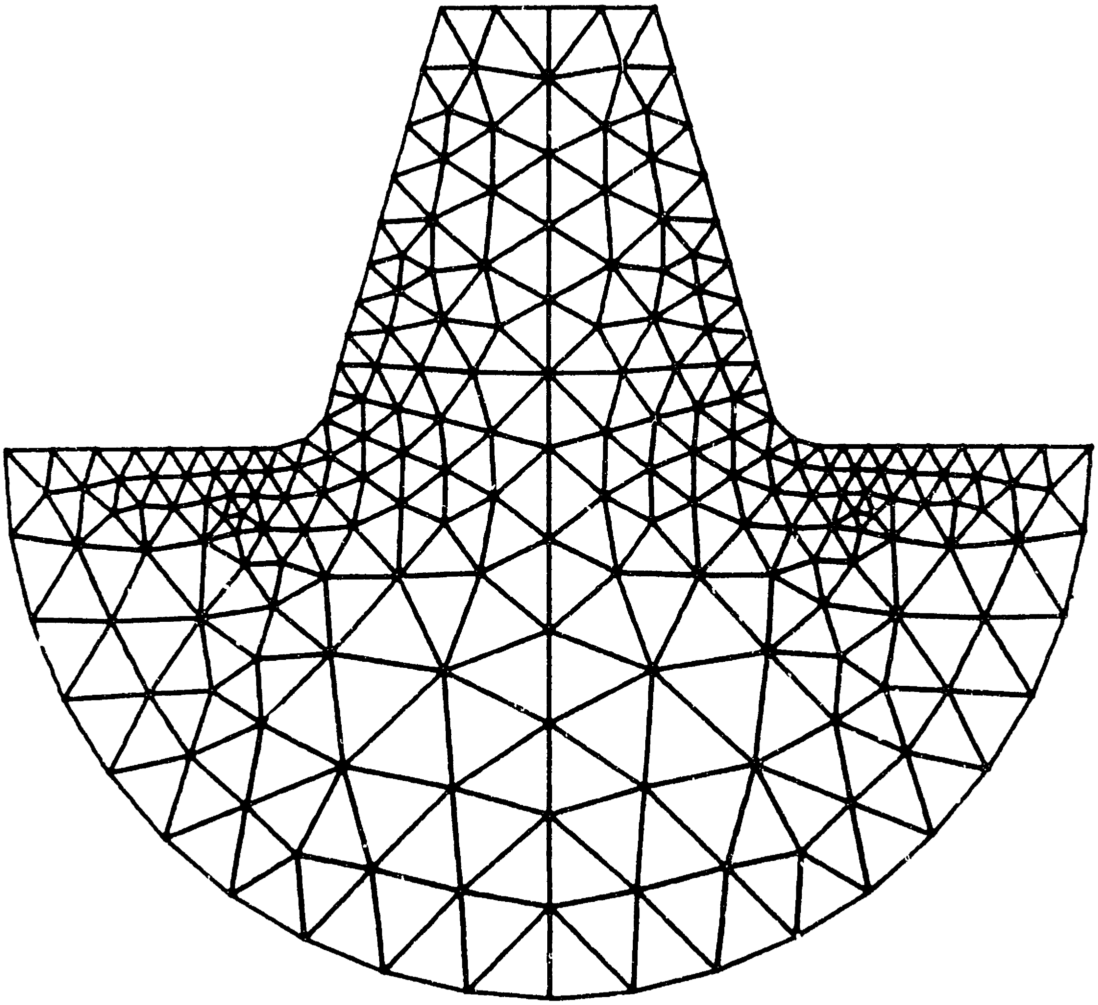


No. of iterations for smoothing :

N(1) = 3,   N(2) = 4

N(3) = 4,   N(4) = 3

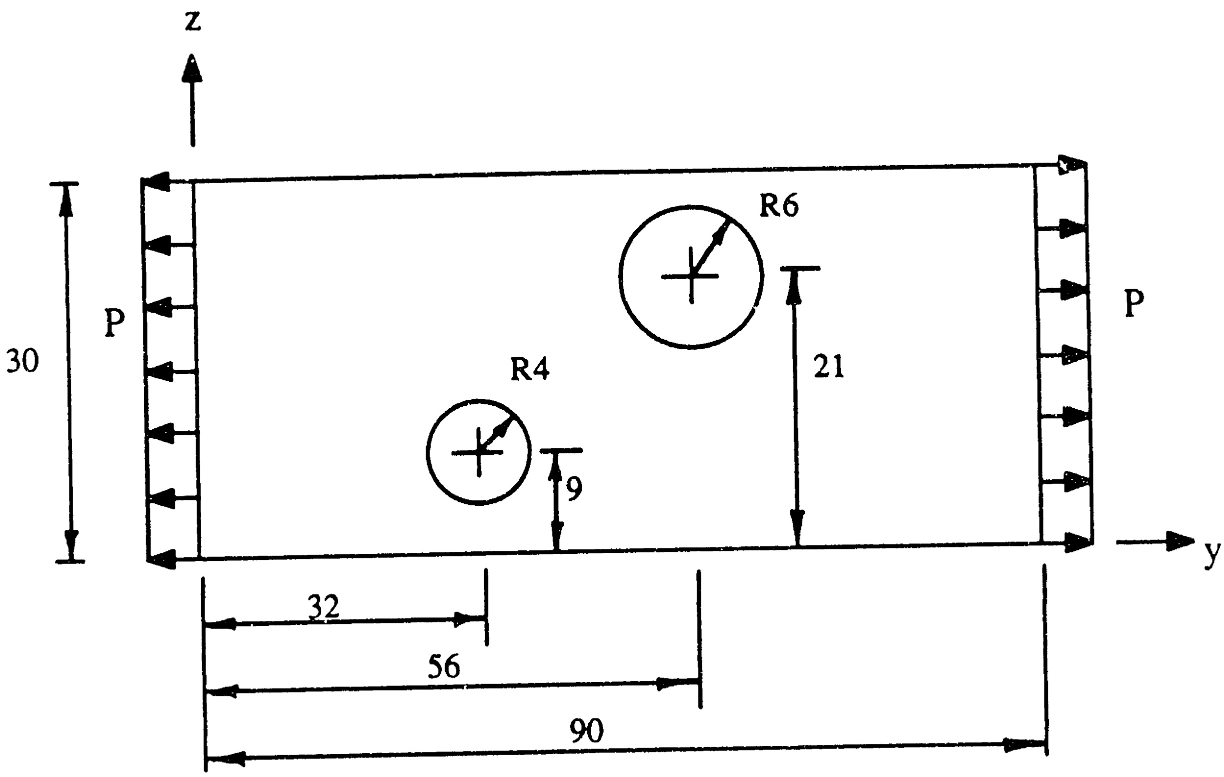
Figure 2-14 a): Triangulation on a complex domain



384 elements

833 nodes

Figure 2-14 b): Results of triangulation



(unit ; cm)

P = 100 Mpa

E = 207000 Mpa

$\nu = 0.3$

thickness = 1

Figure 2-15 a): Plate with multiple holes

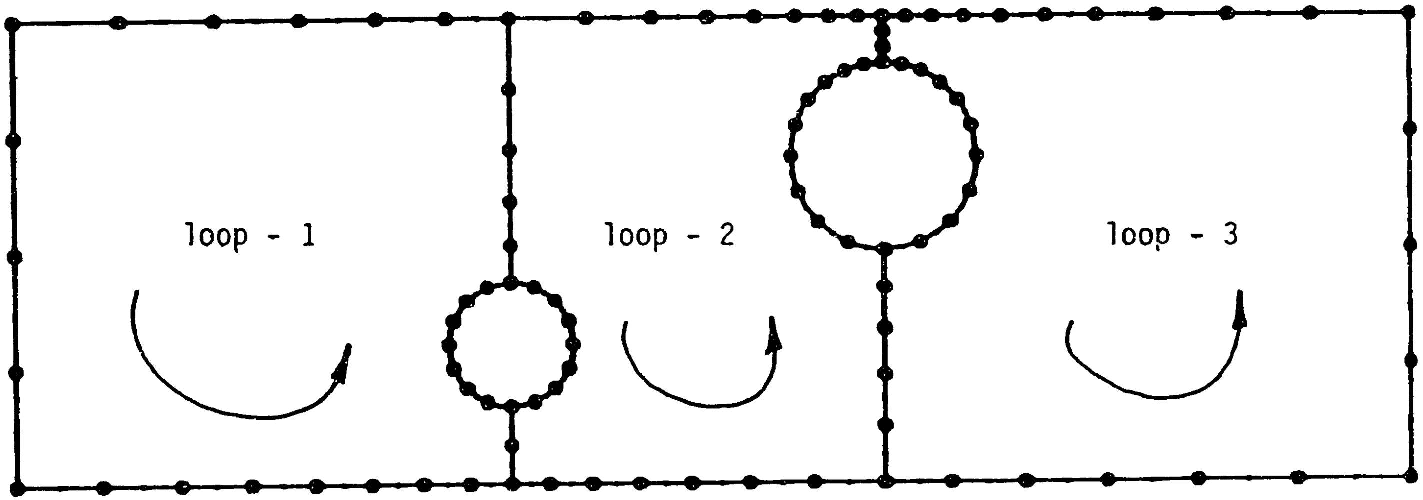
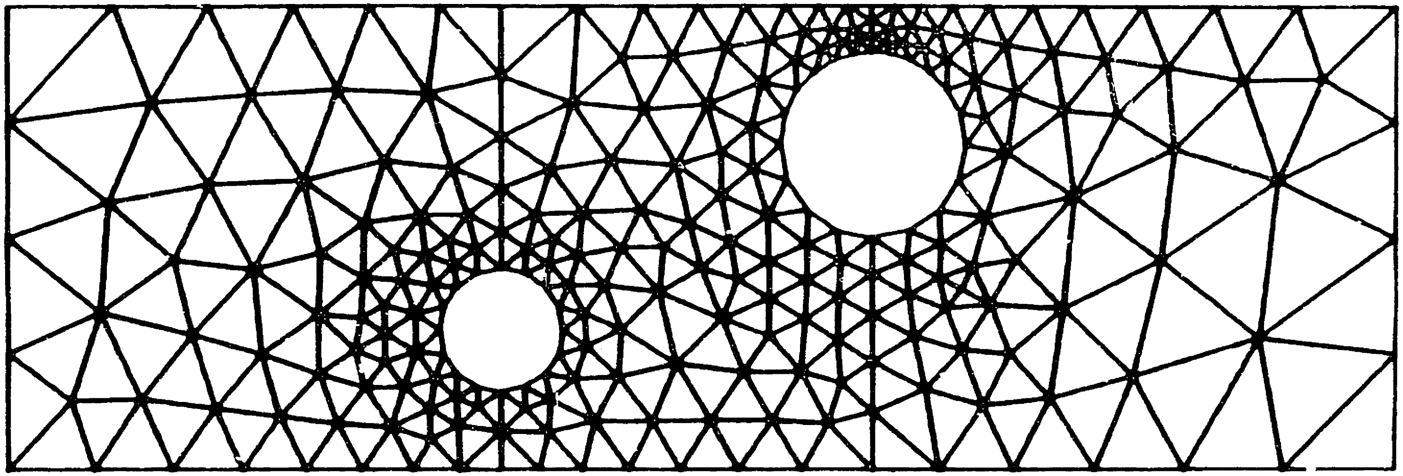


Figure 2-15 b): Key nodes distribution



No. of iterations for smoothing : N(1) = 3, N(2) = 2, N(3) = 4

429 elements, 946 nodes

Figure 2-15 c): Results of triangulation

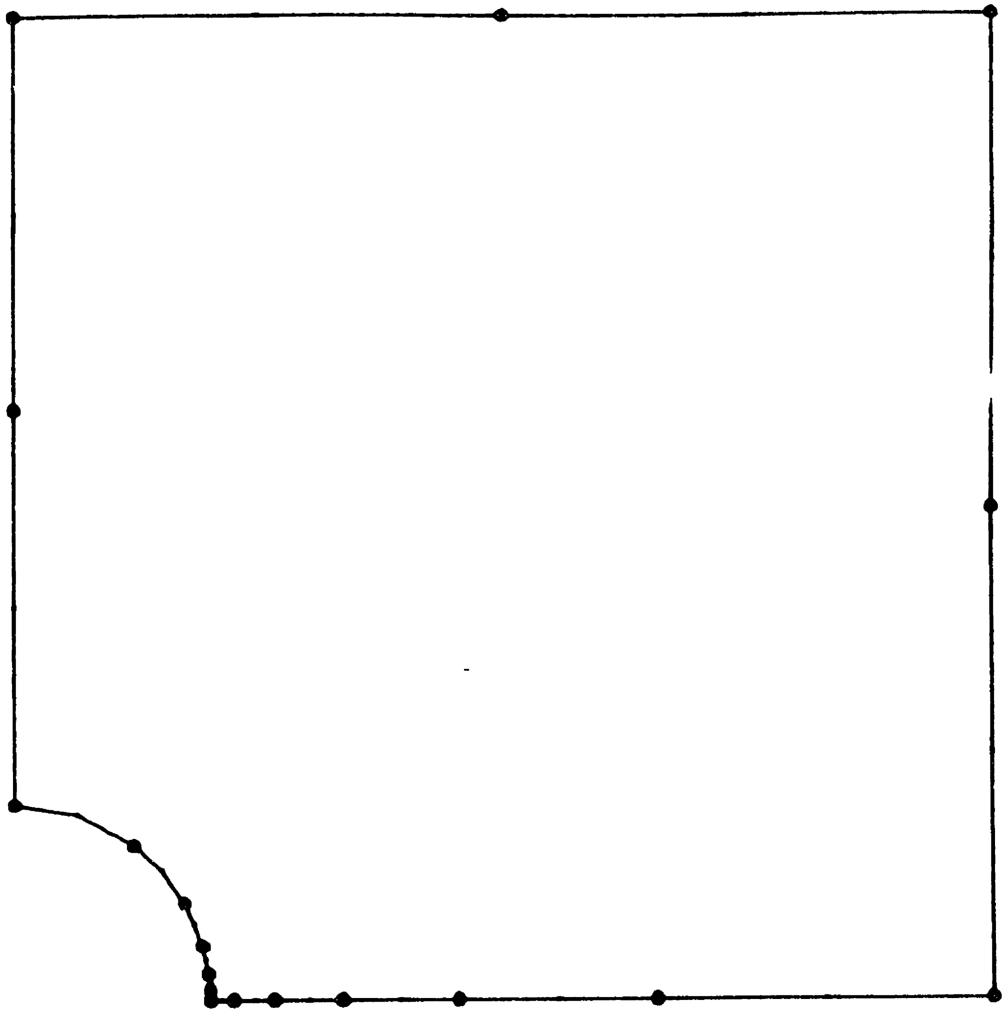
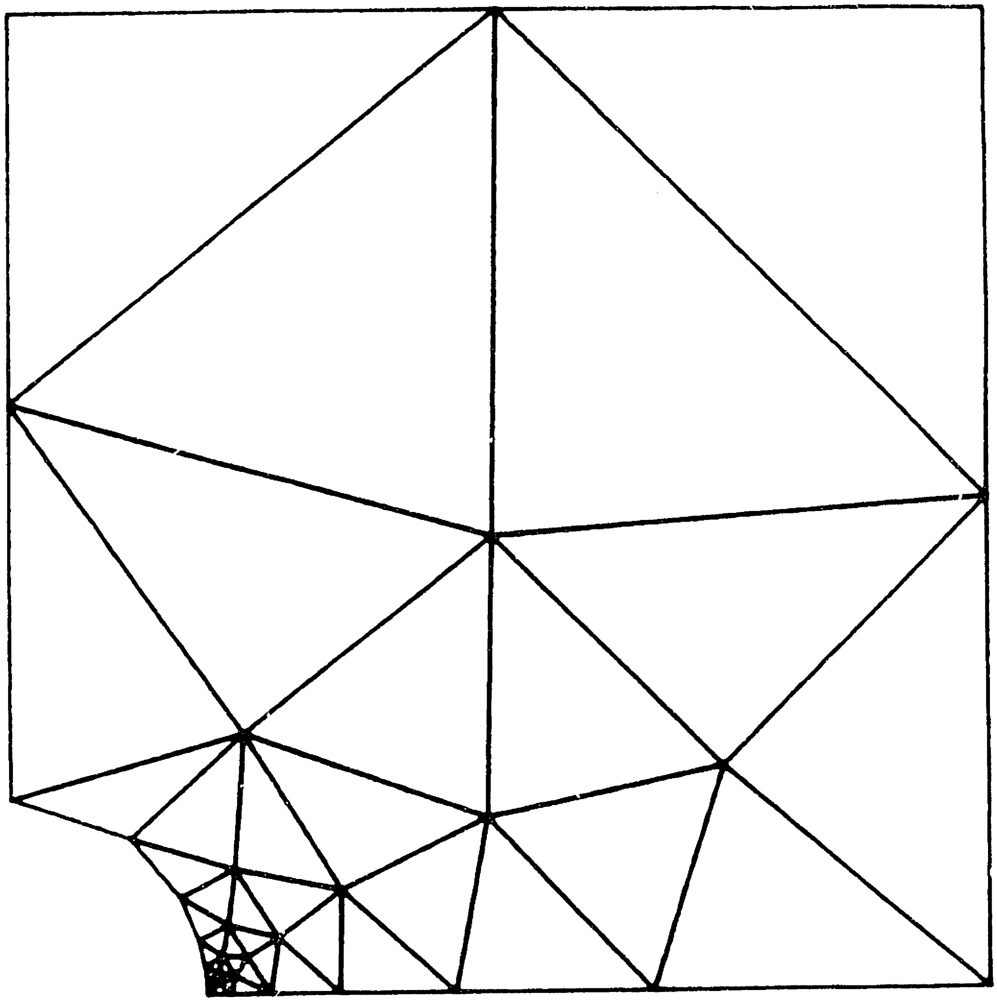


Figure 2-16 a): Key nodes distribution for a plate with a hole



After 6 iterations of smoothing :

42 elements, 103 nodes

Figure 2-16 b): Mesh for a plate with a hole with high local mesh density

The triangulation scheme developed in this research has a certain directionality in the mesh generation. There are two reasons for this phenomenon. One reason is due to the counter-clockwise direction in which the key nodes are ordered. The other reason is that the basic operations are performed one after another, which changes the loop-boundary conditions after every operation. Hence, the loop-boundary may not be symmetric during the triangulation process, although it is symmetric initially.

2.2.2 Error Analysis

Error analysis has been recognized as an essential task in assessing the quality of finite element solutions and for the adaptive refinement process to obtain a near-optimal mesh. One key issue in error analysis is how to define a good error indicator. A good error indicator should satisfy the following conditions.

1. It should be able to indicate the amount of error involved in the finite element solutions.
2. It should be computationally fast to calculate compared with the efforts involved in the analysis itself.
3. It should be easy to implement into an existing code.

Since the pioneering work in the convergence of finite element discretizations by Strang [52], much research effort has been directed on this subject during the last decade. Among those who contributed greatly, Babuska [5-9][11] has paved the way in this field with his works on error estimates and related mathematical studies. The error indicator given below is not far from those earlier proposed and is a variation of a theoretical error indicator, based on our computational experiences.

The error in finite element analysis can be taken as any measure between the exact solution and the finite element solution; we can use the displacements, stresses, strains as energy to measure the error. Among many possible error sources, we consider only the discretization error. Since the exact solutions are not generally known, certain features of the exact solutions must be used to indicate the error, or the exact solutions need be estimated to compute the error involved. Therefore, two different approaches are employed in deriving the error indicators.

One approach is to use that for the exact solution, the error involved in satisfying the equilibrium equations is zero. Previous works based in this approach include those of Babuska and Rheinboldt [5-7], Zienkiewicz and his coworkers [8-9], and Itoh and Wilson [10]. When using the equilibrium equations to obtain an error indicator, two sources of error should be considered: the body force residual error in each element domain and the traction jumps between adjacent elements. The computations of body force residuals which involve the 2nd derivatives of displacements and that of traction jumps between adjacent elements are not straightforward tasks.

Another approach is to estimate the exact solution using a certain algorithm such as a projection process and compute the error indicators using this estimator and the finite element solution. This approach has been studied largely due to the fact that the computation of body force residuals and traction jumps can be complex tasks [12][21].

In our research, we prefer the former approach, because we do not want to include any assumption in deriving the error indicators and, in our view, the computation of body force residuals and traction jumps is manageable.

Recent works of Babuska [11] show that in case of bilinear elements the traction jumps along the element boundaries are dominant in the total error, while in quadratic elements the body force residual error provides the dominant part. Our experiences with the quadratic elements lead to similar observations as described below.

Consider the following elasticity problem shown in Figure 2-17. The equilibrium equation is found to be

$$R_i = \tau_{ij,j} + f_i^B = 0 \qquad in \quad \Omega \qquad (2\text{–}10)$$

with the boundary conditions

$$T_i = \tau_{ij}n_j - t_i^S = 0 \qquad on \quad \Gamma_t$$

$$u = u_0 \qquad on \quad \Gamma_u$$

For each finite element, two sources of error are written as

$$R_i^{FE} = \tau_{ij}^{FE}{}_{,j} + f_i^B \neq 0 \qquad in \quad \Omega_m \qquad (2\text{–}11)$$

$$T_i^{FE} = \tau_{ij}^{FE}n_j - t_i \neq 0 \qquad on \quad \Gamma_m$$

where τ_{ij}^{FE} represents finite element solution anu t_i is the exact unknown force transmitted through the boundary of an element.

Using these two error sources, let us define the body force residual error, $F_i^{\Omega_m}$, and the traction residual error, $F_i^{\Gamma_m}$, in their force unit as

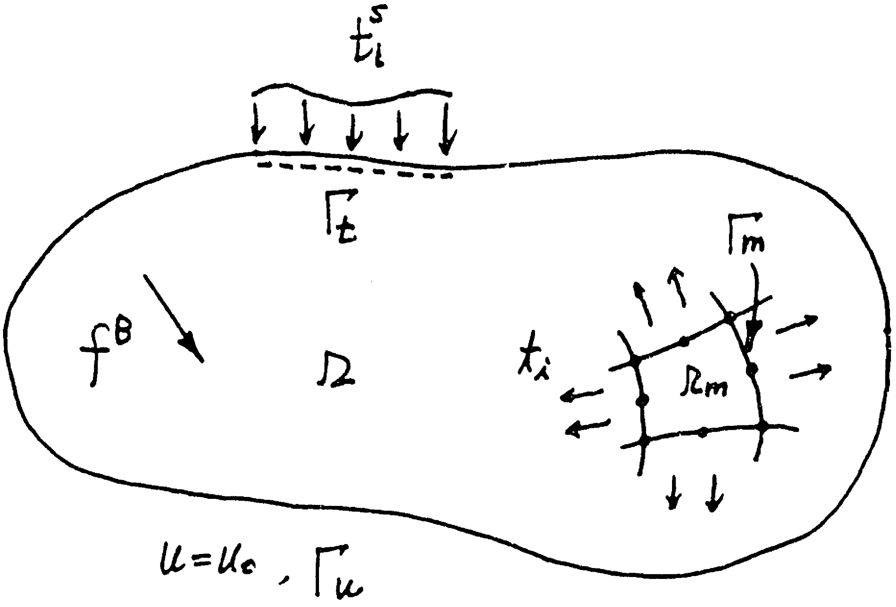


Figure 2-17: Equilibrium state of an elastic body

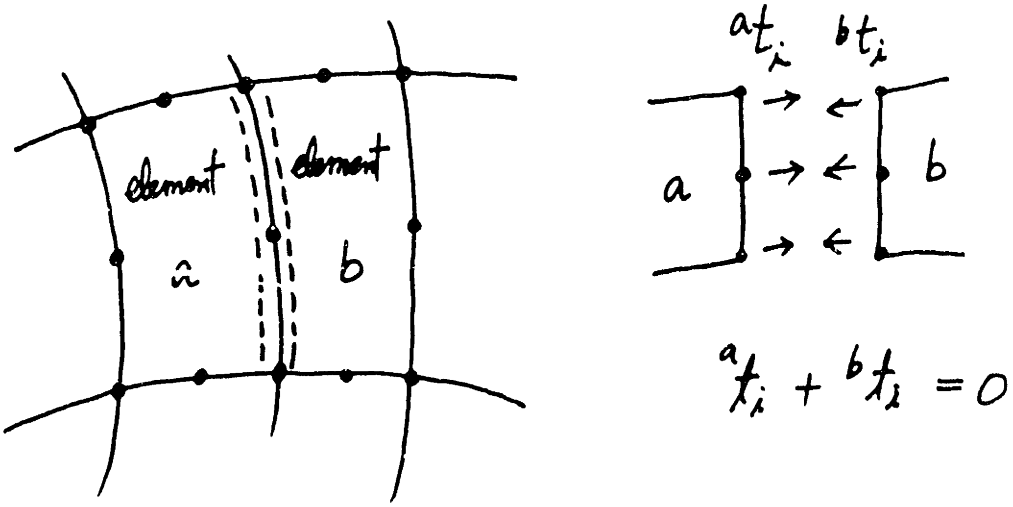


Figure 2-18: Traction jumps between adjacent elements

$$F_i^{\Omega_m} = \int_{\Omega_m} R_i^{FE} \, d\Omega_m \qquad (2\text{–}12)$$

$$F_i^{\Gamma_m} = \int_{\Gamma_m} T_i^{FE} \, d\Gamma_m$$

One interesting relationship can be obtained by using the divergence theorem,see Appendix A for the details.

$$F_i^{\Omega_m} = F_i^{\Gamma_m} \qquad (2\text{–}13)$$

Equation (2-13) shows that the integral of the body force residual error is equal to that of the traction residual error.

Generally, we cannot compute the traction residual error directly, because the exact force transmitted through the element boundary, t, is not known. Instead, the traction jumps between adjacent elements can be used to estimate the traction residual error if a proper error allocation algorithm is available.

Consider two adjacent elements *a*, *b* in Figure 2-18. The traction jump, TM_i, between two elements *a* and *b* is composed of two contributions, ${}^aT_i$ and ${}^bT_i$ such as

$$\begin{aligned} TM_i &= ({}^a\tau_{ij}^{FE} - {}^b\tau_{ij}^{FE}) \cdot n_j^a = {}^a\tau_{ij}^{FE} \cdot n_j^a + {}^b\tau_{ij}^{FE} \cdot n_j^b \\ &= ({}^a\tau_{ij}^{FE} \cdot n_j^a - {}^at_i) + ({}^b\tau_{ij}^{FE} \cdot n_j^b - {}^bt_i) \qquad (2\text{–}14) \\ &= {}^aT_i + {}^bT_i \end{aligned}$$

As a simple error allocation algorithm, the traction jump between adjacent elements is equally distributed to each element for convenience. For our numerical use, the two error indicators in the force norm are defined as

$$F_{\Omega_m} = \int_{\Omega_m} (R_y{}^2 + R_z{}^2)^{1/2} \, d\Omega_m \qquad (2\text{–}15a)$$

$$F_{\Gamma_m} = \frac{1}{2}\int_{\Gamma_m} (TM_y{}^2 + TM_z{}^2)^{1/2} \, d\Gamma_m \qquad (2\text{–}15b)$$

Another common measure is the error in the energy norm,

$$\|\mathbf{e}\|^2 = \int_{\Omega} \mathbf{e}^T (\mathbf{B}^T \mathbf{C} \, \mathbf{B} \, \mathbf{e}) \, d\Omega \qquad (2\text{–}16)$$

where the error, e, is defined as the difference between the exact displacement $\mathbf{u}$ and the approximate solution $\bar{\mathbf{u}}$

$$\mathbf{e} = \mathbf{u} - \bar{\mathbf{u}} \qquad (2\text{–}17)$$

In the above, the matrix operator $\mathbf{B}$ defines the strain $\bar{\varepsilon}$ as

$$\bar{\varepsilon} = \mathbf{B}\,\mathbf{u} \qquad (2\text{–}18)$$

and the material matrix $\mathbf{C}$ defines the stress as

$$\sigma = \mathbf{C} \cdot \bar{\varepsilon} \qquad (2\text{–}19)$$

Babuska and Rheinboldt [6] have proved that there exist constants $k_2 \geq k_1 > 0$ such that

$$k_1\hat{\varepsilon}^2 \leq \|e\|^2 \leq k_2\hat{\varepsilon}^2$$

and (2–20)

$$\hat{\varepsilon}^2 = \sum_{m=1}^{N} \hat{\varepsilon}_m{}^2 = \sum_{m=1}^{N} (C_1 h_m{}^2 \int_{\Omega_m} R^2\, d\Omega + C_1 h_m \int_{\Gamma_m} J^2\, d\Gamma)$$

where R is the residual and J is the inter-element traction jump. In order to see which term is dominant in equation (2-20) in the case of quadratic elements, we will consider the following terms.

$$E_{\Omega_m} = \frac{h_m{}^2}{C}\int_{\Omega_m} (R_y{}^2 + R_z{}^2)\, d\Omega_m \qquad (2\text{–}21a)$$

$$E_{\Gamma_m} = \frac{h_m}{2C}\int_{\Gamma_m} (TM_y{}^2 + TM_z{}^2)\, d\Gamma_m \qquad (2\text{–}21b)$$

where h_m is the characteristic length of an element and C is the Young's modulus.

The above four error indicators in equation (2-15) and (2-21) are non-dimensionalized with respect to the total external force and the total strain energy respectively as follows.

$$f_{\Omega_m} = \frac{F_{\Omega_m}}{F_{external\, force}} \qquad (2\text{–}22a)$$

$$f_{\Gamma_m} = \frac{F_{\Gamma_m}}{F_{external\, force}} \qquad (2\text{–}22b)$$

and

$$e_{\Omega_m} = \frac{E_{\Omega_m}}{E_{total\ strain\ energy}} \tag{2-23a}$$

$$e_{\Gamma_m} = \frac{E_{\Gamma_m}}{E_{total\ strain\ energy}} \tag{2-23b}$$

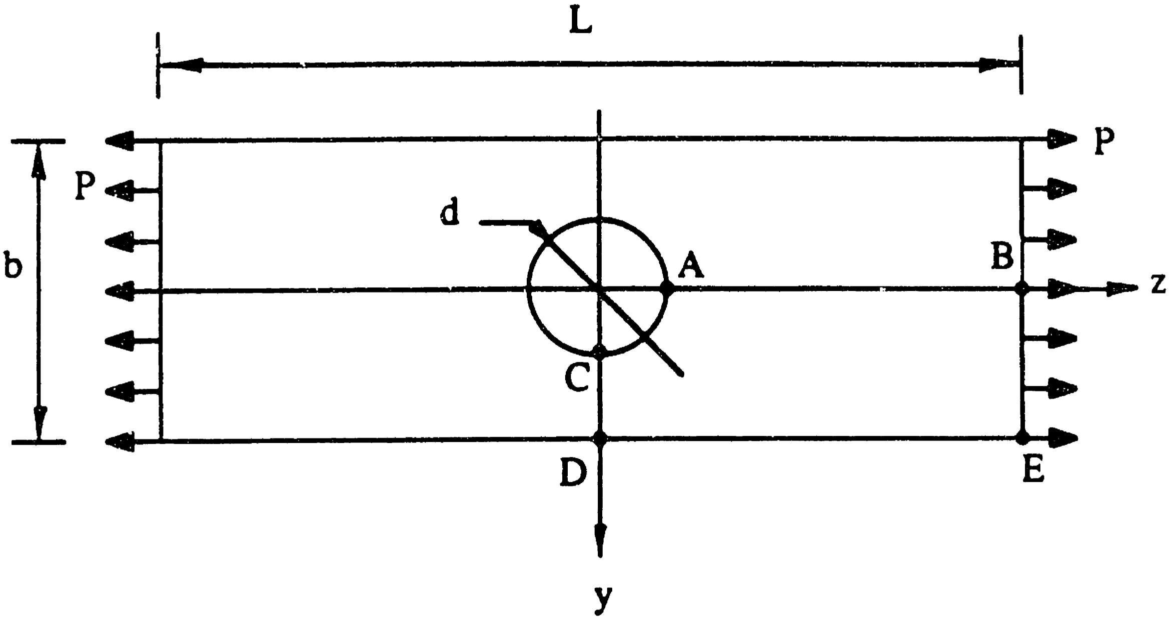
The error indicators in equation (2-22) and (2-23) are computed for the examples in Figure 2-19 to Figure 2-21. Only 8-node quadrilateral elements are used for the analysis. The results for the calculation of the error indicators are summarized in Table 2-1.

One interesting point to notice in Table 2-1 is that the traction jump error and the body force residual error in the force norm are almost the same in their magnitudes, while in the energy norm, the body force residual error is the dominant one.

The residual error in the energy norm is about twice larger than the traction jump error for the problem in Figure 2-19, while for the problems in Figure 2-20 and 2-21, the residual term is about 50 - 100 times larger than the traction jump term. Hence, it can be generalized that the body force residual in the energy norm is much larger than the traction jump residual, especially for problems with high stress concentration.

In our numerical tests, the error indicator in the energy norm is found to be more efficient than the one in the force norm for the adaptive refinement process.

Therefore, the best choi:e of the error indicator for an adaptive refinement process using quadratic elements appear to be the body force residual in the energy norm as in equation (2-21a).



$L = 56$ mm $\qquad E = 7.0\times10^{4}$ N-m

$b = 20$ mm $\qquad \nu = 0.25$

$d = 10$ mm $\qquad P = 25.0$ N/mm^{2}

$h = 1$ mm (thickness)

Figure 2-19 a): Plate with a hole in tension

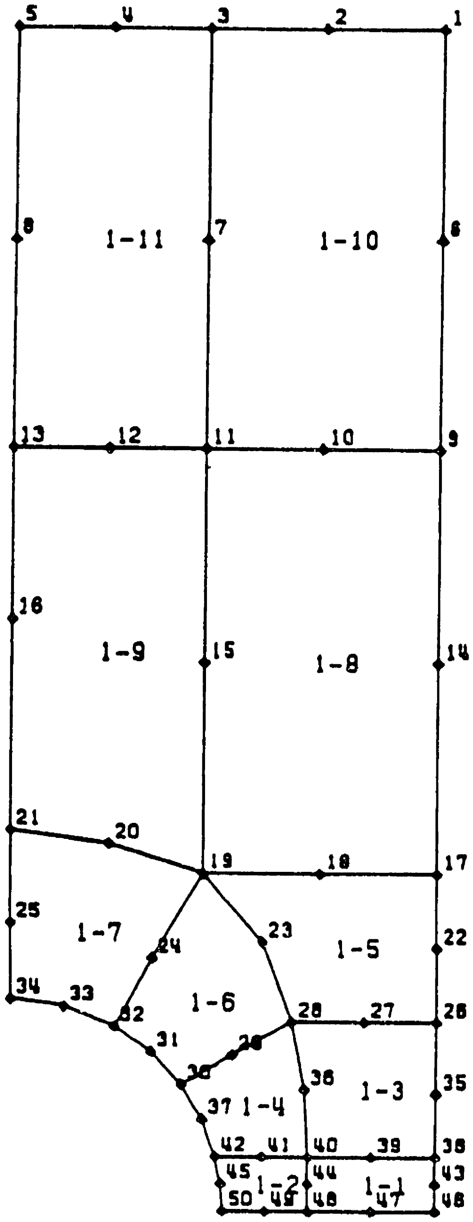
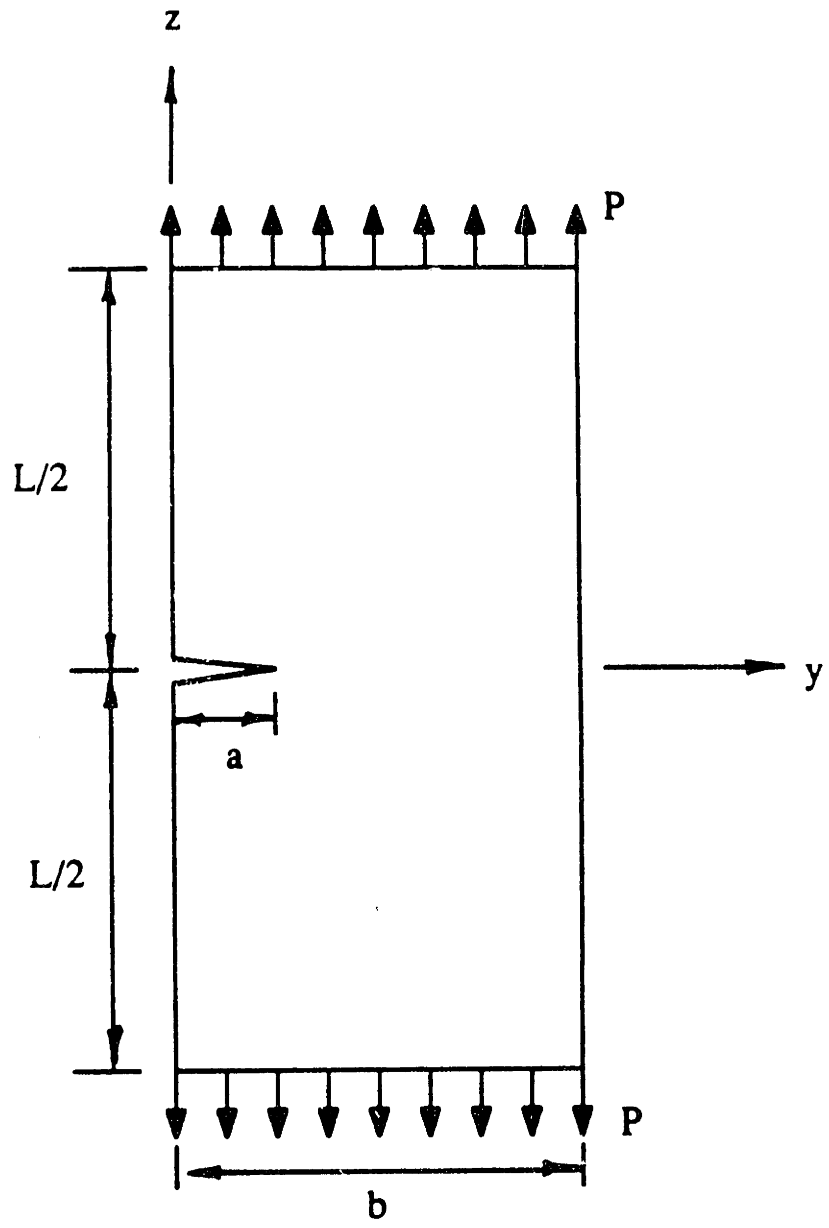


Figure 2-19 b): 11 element model



L = 0.32 m	E = 2.07×10^{11} N/m^2
b = 0.16 m	ν = 0.29
a = 0.04 m	P = 1000 N/m^2
h = 1.0 m (thickness)	

Figure 2-20 a): Crack tip problem

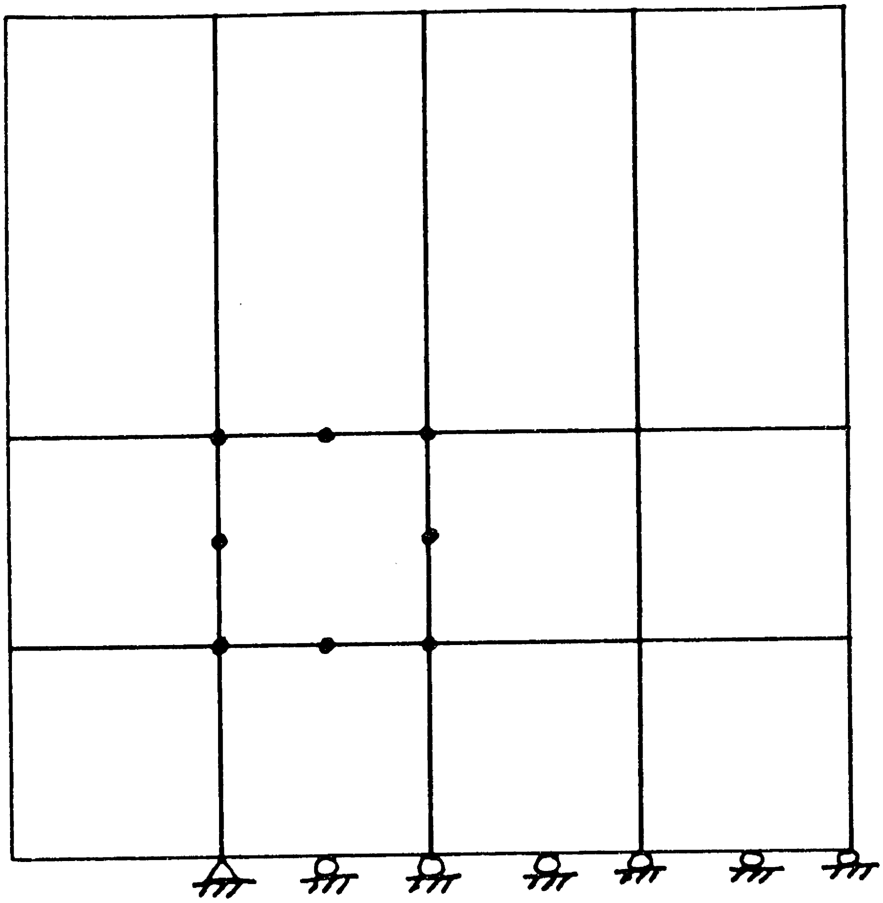
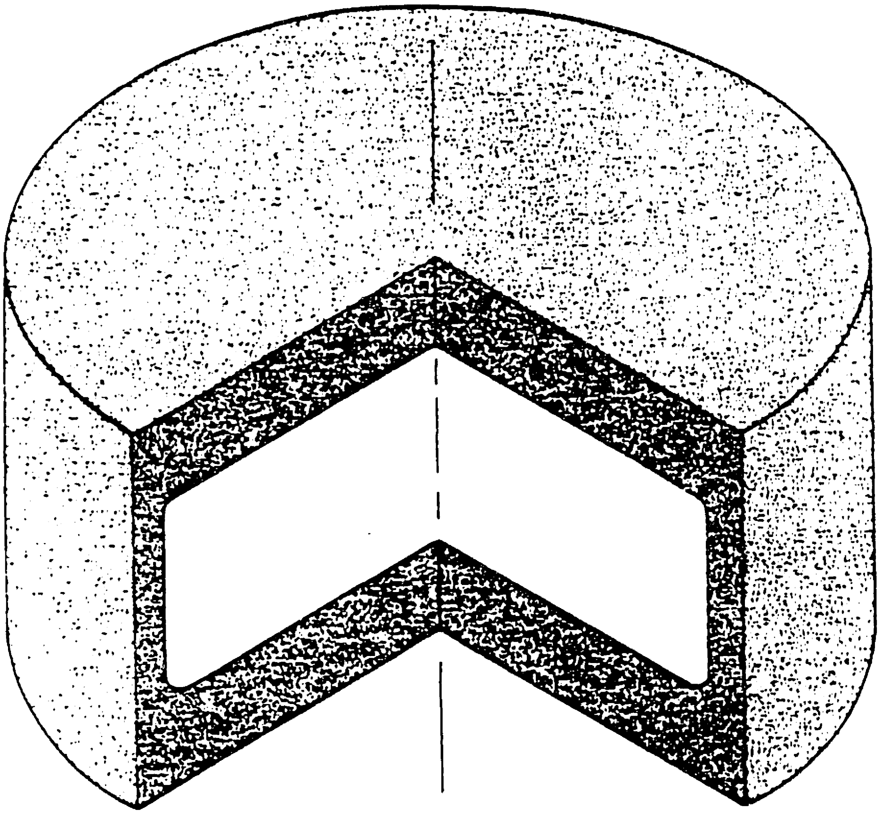
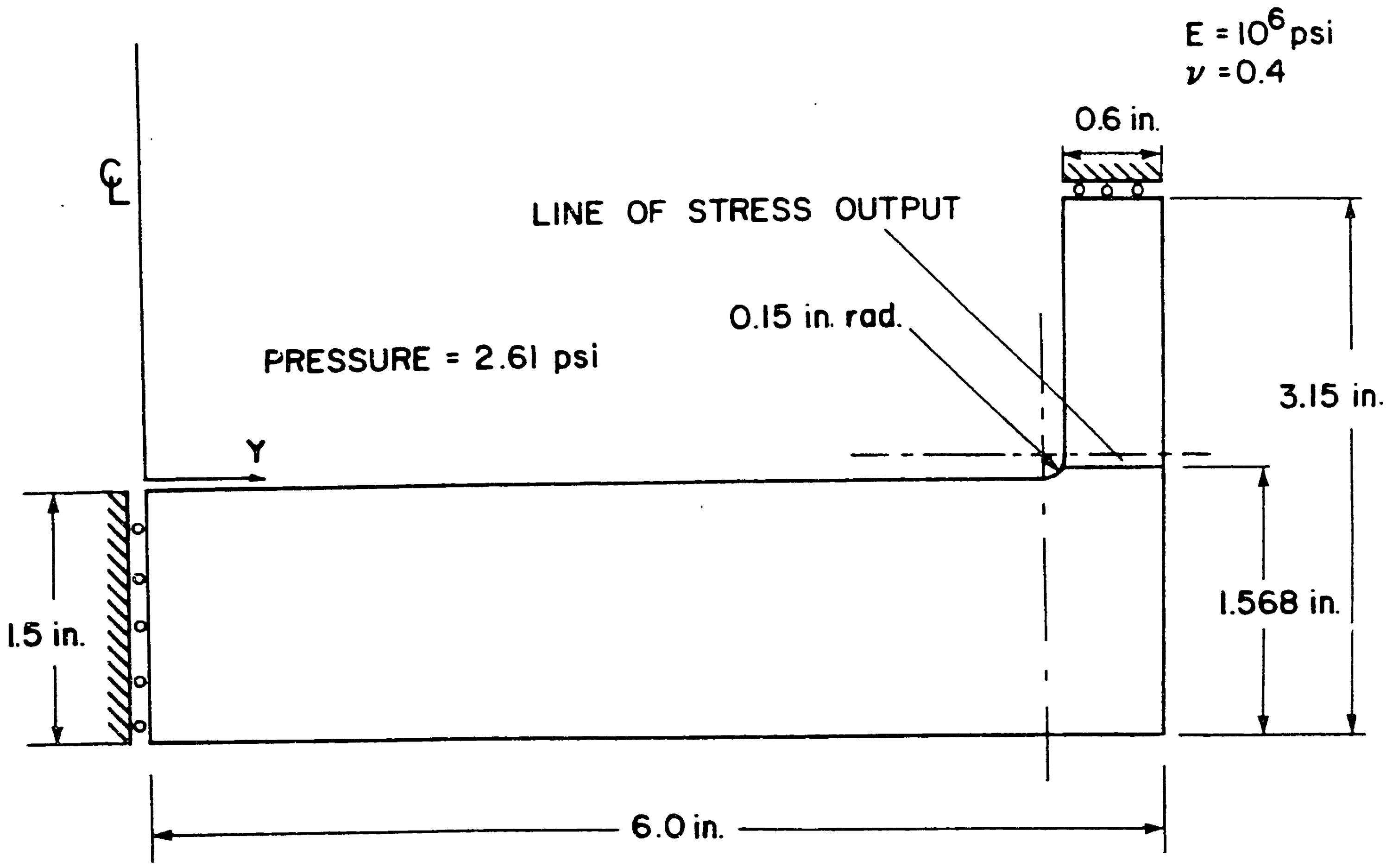


Figure 2-20 b): 12 element model

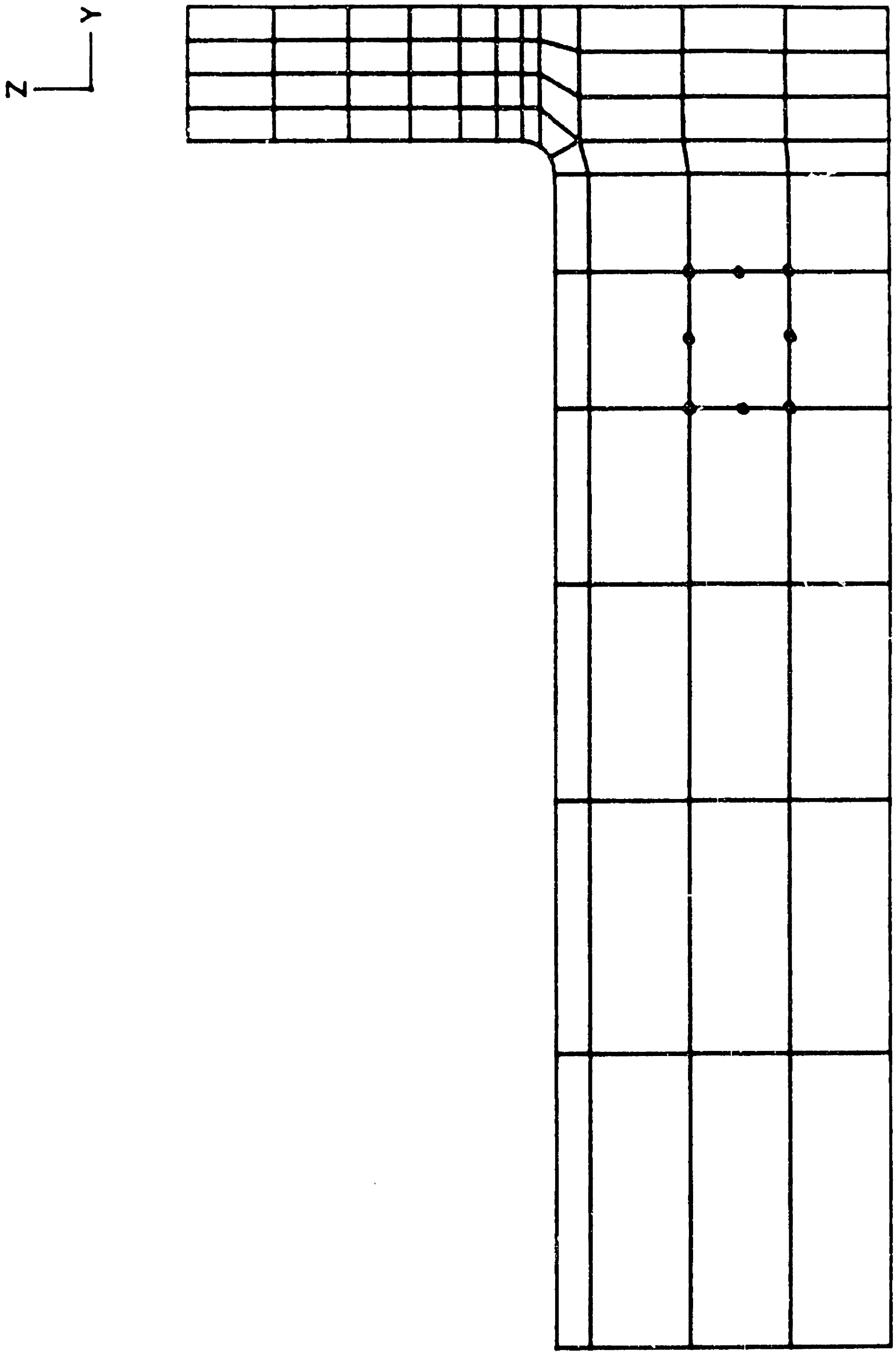


(a) Schematic drawing [61]

Figure 2-21: Pressure vessel problem



(b) Dimensions



(c) 69 element model

Table 2-1. Comparison of the error indicators

	Non-dimensionalized (w.r.t.force)		Non-dimensionalized (w.r.t.energy)	
	Traction $\sum_m f_{\Gamma_m}$	Residual $\sum_m f_{\Omega_m}$	Traction $\sum_m e_{\Gamma_m}$	Residual $\sum_m e_{\Omega_m}$
A. Structure with a hole (Figure 2-19)				
1) Plane stress (t=1)	1.03	1.21	0.0692	0.14
2) Plane strain	1.10	1.27	0.0829	0.169
3) Axisymmetric	0.49	0.533	0.0202	0.0424
B. Structure with a crack tip (Figure 2-20)				
1) Plane strain	0.959	0.949	0.00777	0.593
2) Plane stress (t=0.01)	0.950	0.902	0.00747	0.506
3) Axisymmetric	0.179	0.172	0.00133	0.13
C. Structure of a vessel form (Figure 2-21)				
1) Axisymmetric	3.50	4.71	0.00779	0.424
2) Plane strain	4.32	5.86	0.00288	0.148
3) Plane stress (t=0.1)	2.99	4.20	0.00118	0.053

$$\eta_m{}^0 = \frac{h_m{}^2}{E}\int_{\Omega_m}(\sum_i R_i{}^2)\, d\Omega_m \tag{2-24}$$

Using the error indicator in equation (2-24), we observe the following in terms of efficiency. In most cases, we are interested in the solution of a stress concentration and the element size in this region should become more rapidly smaller than in the other regions during the refinement process. This aim in the refinement is enhanced by introducing the relaxation factor $1/h$: we multiply the above error indicator by $1/h$. Namely, this way we scale up the error indicator of the small sized elements compared with the larger elements, consequently refinement is more concentrated in the region of the small sized elements. The result is that better solutions can be obtained at stress concentrations with little sacrifice on the overall accuracy. Therefore, the following error indicator will be used in our discussion,

$$\eta_m = \frac{h_m}{E}\int_{\Omega_m}(R_y{}^2 + R_z{}^2)\, d\Omega_m \tag{2-25}$$

where h_m represents the characteristic length of an element, E is Young's modulus, R_y and R_z are the body force residuals, and Ω_m is the volume of an element. The body force residuals, R_y and R_z, for two-dimensional problems are summarized in Appendix B.

As a reference value, the total strain energy of the system has been used with the following modification.

$$U_{reference} = \frac{U_{total}}{L_{system}} \tag{2-26}$$

where $U_{reference}$ is the reference strain energy per unit length, and U_{total} is the total strain energy of the system, and L_{system} is the characteristic length of the system.

As a criterion for further refinement, a non-dimensionalized error indicator for each element, "m", defined as follows is used,

$$\varepsilon_m = \frac{\eta_m}{U_{reference}} \qquad (2-27)$$

The total error indicator of the system is found to be

$$\varepsilon_{total} = \sum_{m=1}^{N} \varepsilon_m \qquad (2-28)$$

Our numerical experiences on this error indicator show that the total error indicator in equation (2-28) is able to indicate the amount of error involved in the finite element solutions approximately.

2.2.3 Adaptive Refinement

In this research, the *h*-version of refinement has been adopted as a grid enrichment method, in which the size of elements is halved by subdividing each element into four subelements. In order to deal with this refinement process automatically, a special data structure is developed which stores and updates all the information about the elements such as the edges and faces and their connectivity.

Most of the adaptive solvers use the tree-structure for the data management [11][53]. In the tree structure, the connectivity data are stored in a compact form and the subdivision of an element can be made simply by extending the tree. On the other hand, since the tree structure does not store the redundant connectivity information, the data management is not a simple task and can only be achieved by extensive use of up-path and down-path searches.

Therefore, in order to eliminate the need for searching, a boundary representation data structure with redundant information is designed in our scheme. The basic idea of the data structure for this purpose is similar to the winged-edge data structure [47-48], which is widely adopted for boundary representation solid modelers in CAD/CAM systems. The winged-edge solid modeler in three dimensional problems stores the data for the faces, edges, and vertices of an object and their connectivity, while in two dimensional problems it stores the data for the edges and vertices of a loop.

In our scheme, the concept of an object/loop in a solid modeler is applied to each finite element, a tetrahedron/triangle respectively. Hence, each tetrahedron element enclosed by four small faces is considered to be a subobject and each triangular element enclosed by three small edges is considered to be a subloop. This modified data structure is called "*dissembled winged-edge/face data structure*", in the sense that it has a winged-edge like data structure for two dimensional problems and a winged-face like data structure for three dimensional problems and the original object/loop is considered to be an assemblage of many subobjects/subloops.

One disadvantage of this data structure is the complexity involved in changing the connectivity information during the refinement process. But this difficulty can be overcome by using our pre-designed refinement unit, in which the change in connectivity data can be well managed. Another disadvantage can be the increased storage requirement due to the redundant information. However,this is not an important problem in practice, because even the increased storage requirement is still less than that of the finite element analysis itself.

Since the refinement process produces *irregular* nodes and edges, a concept not dealt with in previous winged-edge data structures, some new concepts have been introduced to deal with this problem.

A *regular* node is defined as an independent node in terms of degrees of freedom, while an *irregular* node is defined as a dependent node, the movement of which is constrained by independent nodes. Here, a *regular* edge refers to an edge which is composed of three regular nodes, two end nodes and one mid-side node, and there is only one element at each side of the edge except the boundary edges. The boundary edge has one element only on one of its sides. An *irregular* edge refers to an edge which includes irregular nodes and there are more than one element at one side while at the other side there is only one element. The examples of regular and irregular nodes and edges are shown in Figure 2-22.

The designed data structure stores the connectivity between edges, nodes, and elements as shown in Figure 2-23, where E is an edge which is an intersection of two or more triangular elements, N is a node which is an intersection of three or more edges, EL is an element.

The refinement process is performed using the basic refinement unit shown in Figure 2-24. During the refinement process, the compatibility conditions between adjacent elements should also be satisfied by using constraint equations. Therefore, a constraint equation management, i.e. generating, deleting and modifying constraints, is not a simple task. But in order to simplify the problem, a

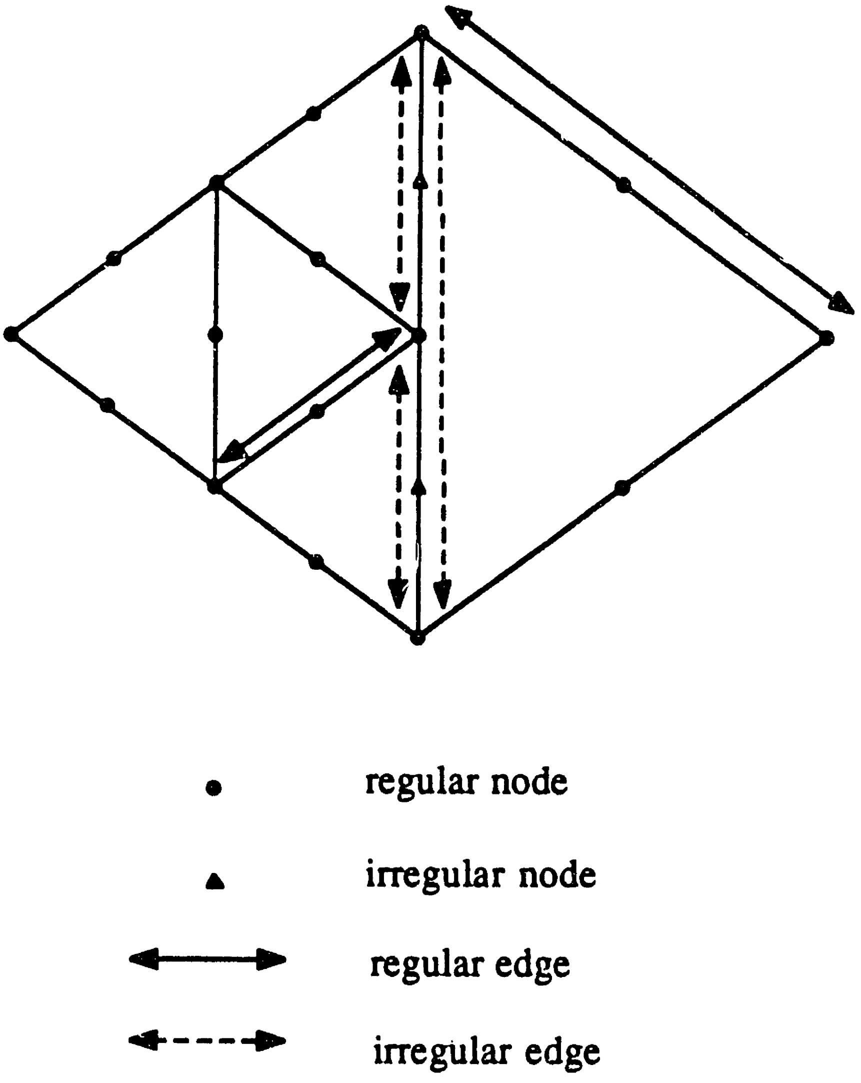
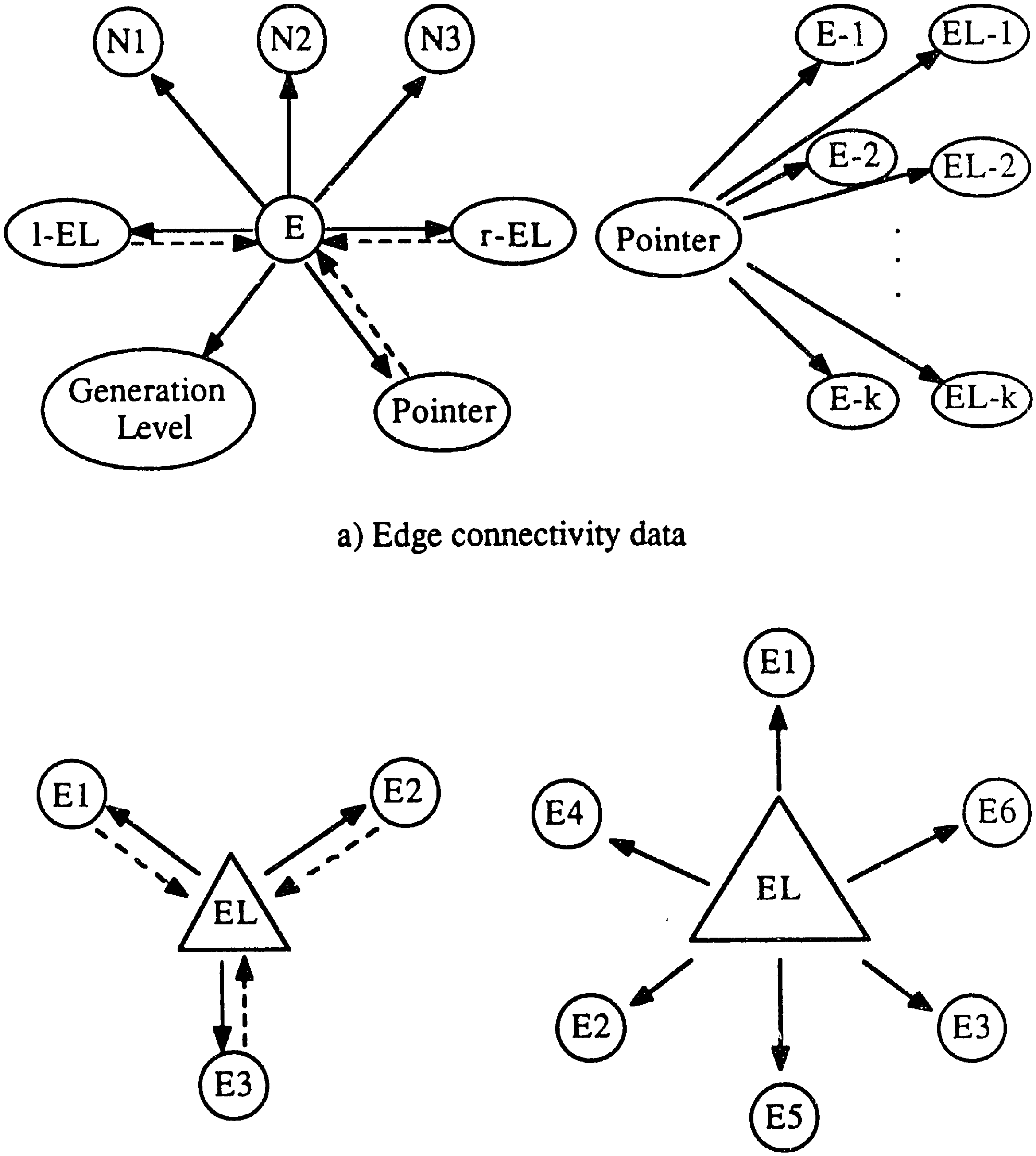


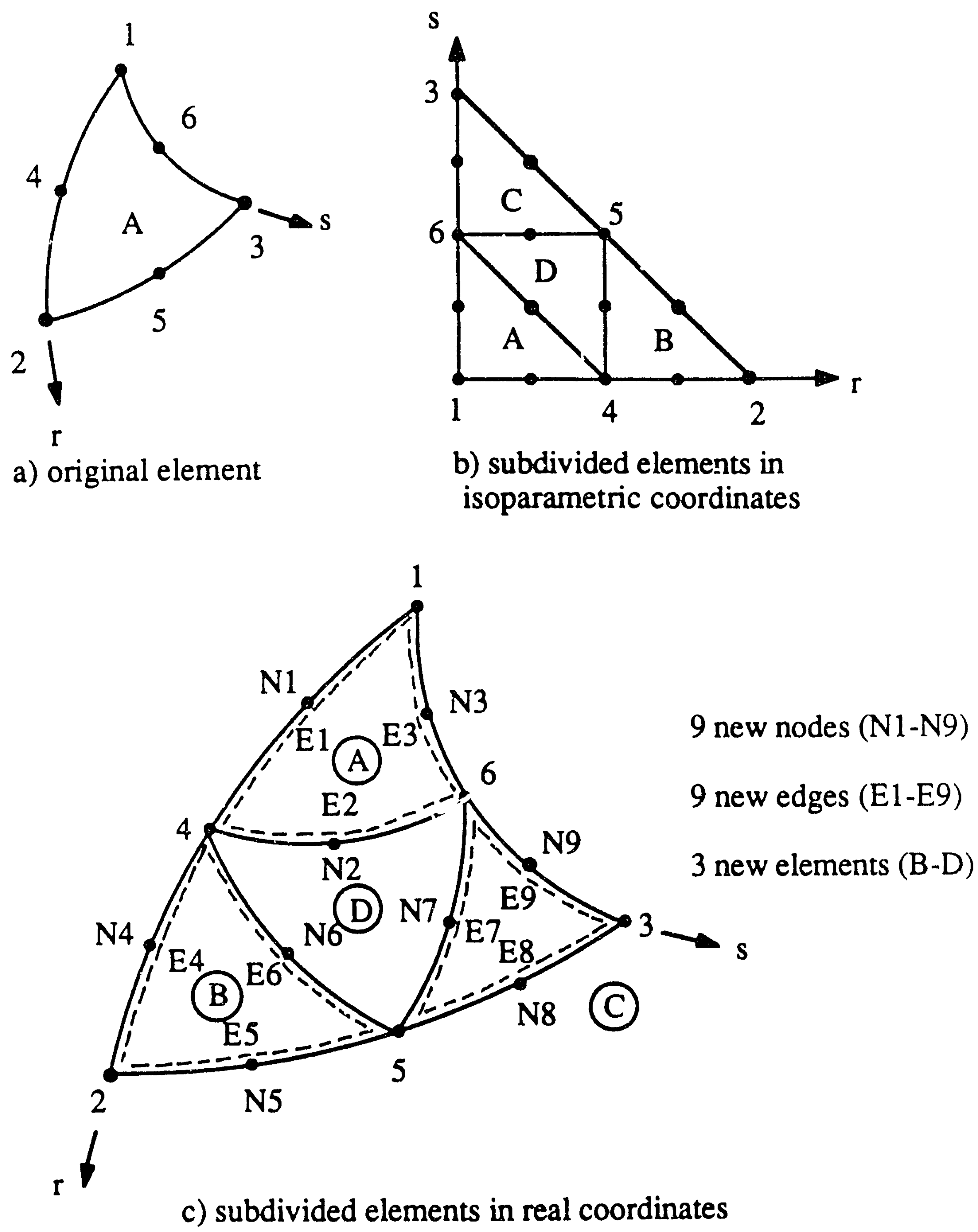
Figure 2-22: Regular and irregular nodes and edges



a) Edge connectivity data

b) Element connectivity data

Figure 2-23: Data structure for refinement process



a) original element

b) subdivided elements in isoparametric coordinates

c) subdivided elements in real coordinates

9 new nodes (N1-N9)

9 new edges (E1-E9)

3 new elements (B-D)

Figure 2-24: Basic refinement unit for a 6-node triangular element

mid-node is always located at the middle of an edge. In fact, this is not a serious restriction at all, because generally a quadratic element performs better when all the mid-nodes are located at the middle of the corresponding edges.

Consider the case shown in Figure 2-25. Each component of displacement for node 4 and node 5 , ${}^4\Delta_i$ and ${}^5\Delta_i$, can be represented in terms of the corresponding components of regular nodes, node 1,2 and 3 as follows.

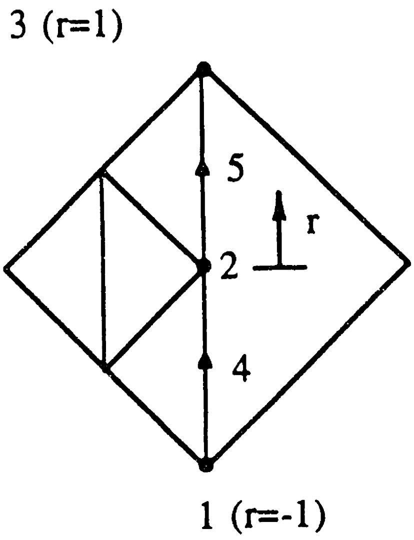
$$\begin{aligned} {}^4\Delta_i &= h_1|_{r=-0.5} \cdot {}^1\Delta_i + h_2|_{r=-0.5} \cdot {}^2\Delta_i + h_3|_{r=-0.5} \cdot {}^3\Delta_i \\ &= \frac{3}{8} \cdot {}^1\Delta_i + \frac{3}{4} \cdot {}^2\Delta_i - \frac{1}{8} \cdot {}^3\Delta_i \qquad (2\text{–}29a) \end{aligned}$$

$$\begin{aligned} {}^5\Delta_i &= h_1|_{r=0.5} \cdot {}^1\Delta_i + h_2|_{r=0.5} \cdot {}^2\Delta_i + h_3|_{r=0.5} \cdot {}^3\Delta_i \\ &= -\frac{1}{8} \cdot {}^1\Delta_i + \frac{3}{4} \cdot {}^2\Delta_i + \frac{3}{8} \cdot {}^3\Delta_i \qquad (2\text{–}29b) \end{aligned}$$

If either node 1 or 3 is an irregular node, the above constraint equations should be modified by substituting the corresponding constraint equations into the above equations. The case when both node 1 and 3 are irregular nodes are excluded in our discussion because of the complexity and rare occurrence.

The constraint equation management scheme is designed to deal with a maximum difference of 2 in the generation level. Thus, up to 4 elements on one side with respect to one element on the other side are allowed as shown in Figure 2-26. If further refinement is necessary and the difference of generation level is already 2, the adjacent element is refined first in order to keep the maximum difference 2.

Consider an example of modifying the constraint equations as shown in Figure 2-27. The constraint equations before refinement of element A are set



$$h_1 = 1/2\, r\,(r-1)$$

$$h_2 = 1 - r^2$$

$$h_3 = 1/2\, r(r+1)$$

Figure 2-25: Example of constraint equations

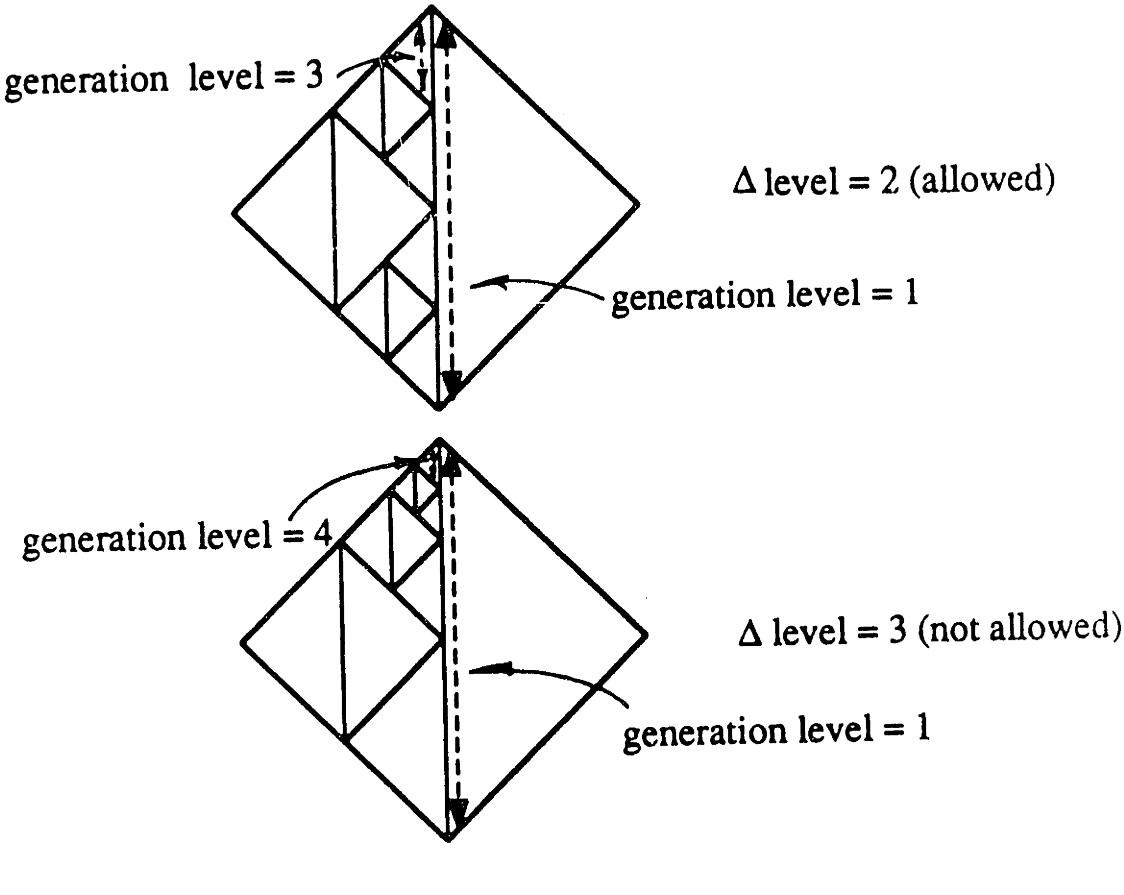
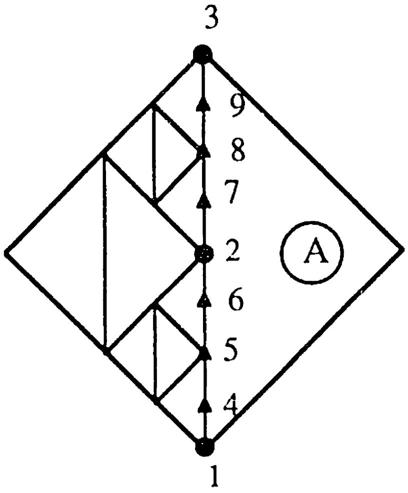
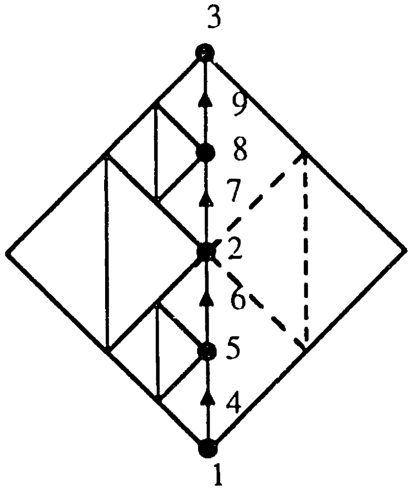


Figure 2-26: Limit of refinement



constraint eqns for nodes	4 5 6 7 8 9
regular nodes used	1 2 3

a) before refinement of element A



constraint eqns for nodes	4 6	7 9
regular nodes used	1 5 2	2 8 3

a) after refinement of element A

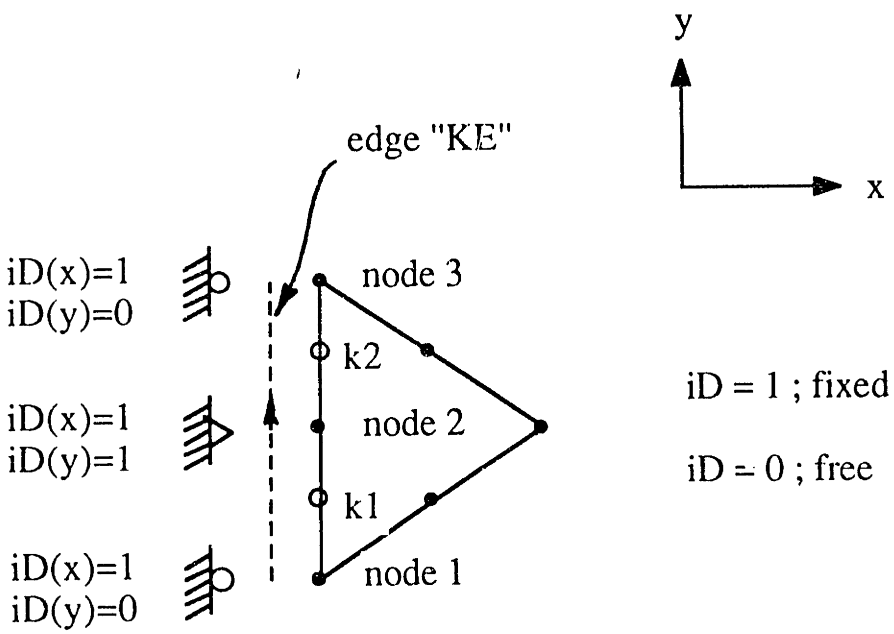
Figure 2-27: Example of modifying constraint equations

up for nodes 4, 5, 6, 7, 8, 9 in terms of regular nodes 1, 2, 3. After refinement, node 5 and node 8 become regular nodes, so the corresponding equations should be removed and the equations for nodes 4,6 and nodes 7,9 should be modified in terms of new regular nodes, nodes 1, 5, 2 and nodes 2, 8, 3 respectively.

If a mid-side node is not located at the middle of an original edge, the newly generated nodes and corresponding constraint equations due to the refinement should be carefully generated in order to satisfy the compatibility conditions between neighbouring elements. But this is not included in the thesis. A typical example of using this scheme would be the case of refining the singular elements used at the crack tip where the mid-node is located at a quarter point of an edge.

In the case of boundary edge refinement, boundary conditions of new nodes should also be generated appropriately. The rules employed for generating the boundary conditions are shown in Figure 2-28. In Figure 2-28, the boundary conditions for nodes k_1, k_2 are generated by using the given boundary conditions on nodes 1,2, and 3.

The example of this operation is shown in Figure 2-29. If there is any pressure load on the refined boundary edge, it should be updated appropriately. In addition to the above process, all the edge connectivity data should also be updated.



```
if ( ID(i,node1).eq.0 .or. ID(i,node2).eq.0 ) then
            ID(i,k1) = 0
else
            ID(i,k1) = 1
endif

if ( ID(i,node2).eq.0 .or. ID(i,node3).eq.0 ) then
            ID(i,k2) = 0
else
            ID(i,k2) = 1
endif

where   i = x, y, z
```

Figure 2-28: Boundary edge refinement

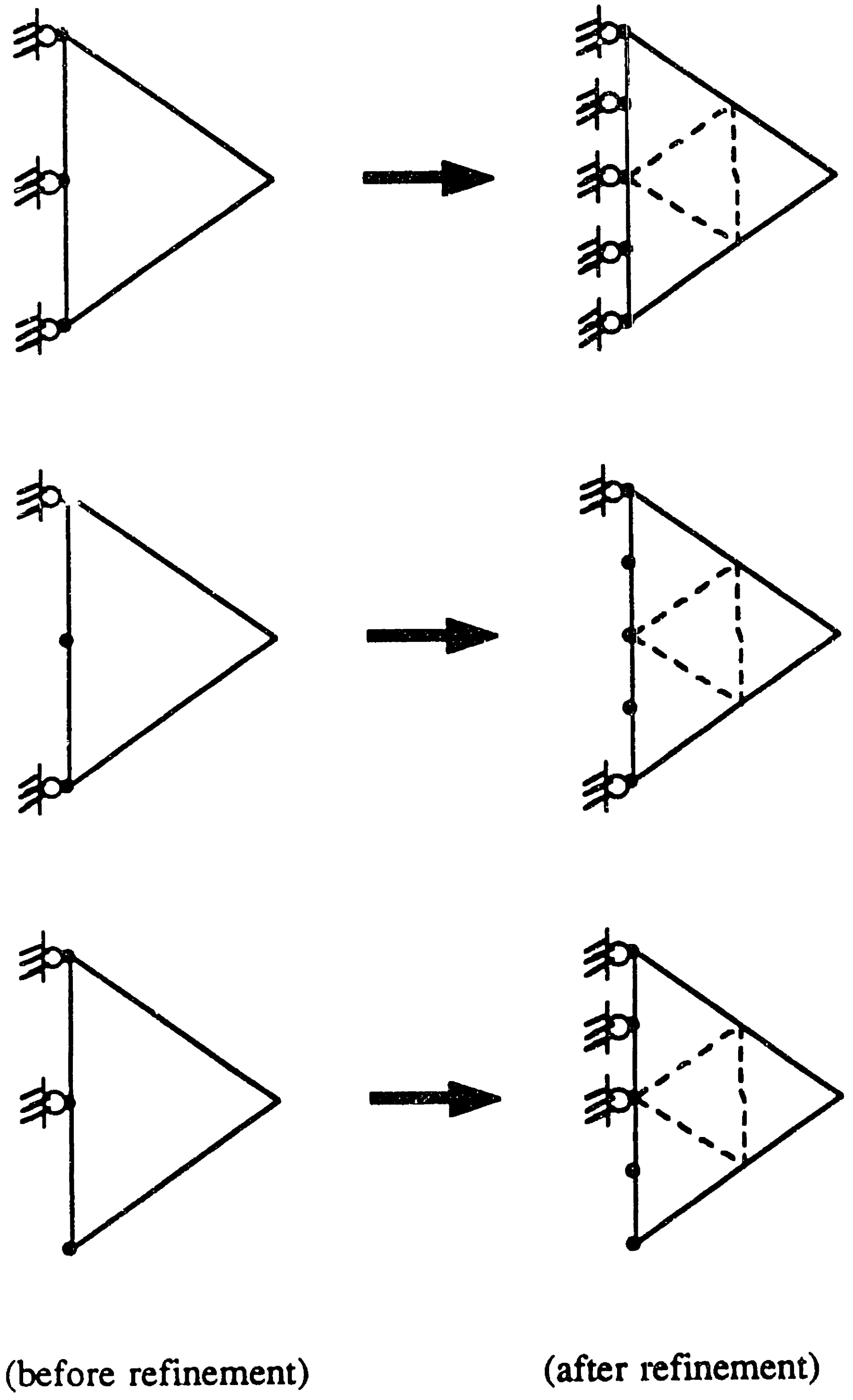


Figure 2-29: Generation of boundary conditions

2.3 Adaptive Refinement for 8-Node Quadrilateral Elements

It is generally accepted that 8-node quadrilateral elements are usually most effective (except in wave propagation using the central difference method with a lumped mass matrix) [54]. However, such an algorithm to construct an 8-node element mesh in any arbitrary domain automatically was not implemented, and a near-optimal mesh can only be obtained in this research through the adaptive refinement process after the initial mesh has been constructed manually.

An 8-node element is designed to have fixed edge directions and the edge numbering scheme is shown in Figure 2-30. The initial data structure for the edge connectivity is automatically set up in the program developed, providing the edge directions between adjacent elements are correctly assigned as shown in Figure 2-31.

Similar to the 6-node triangular element case, the basic refinement unit is designed as shown in Figure 2-32.

The details of the refinement process such as the constraint equation management, boundary condition updating, pressure loading updating and etc. are the same as for the 6-node triangular element case described in Chapter 2.2.

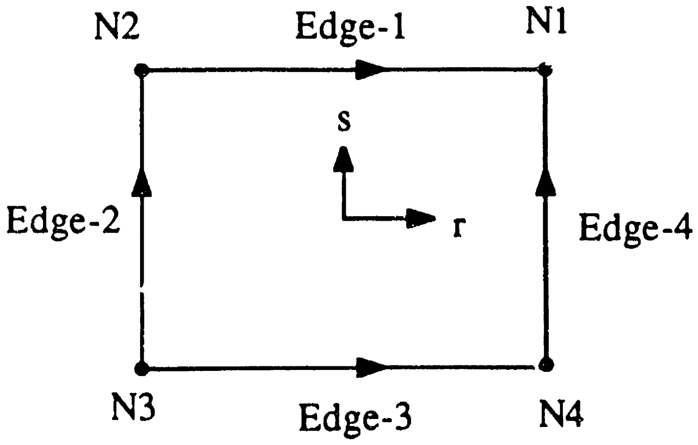
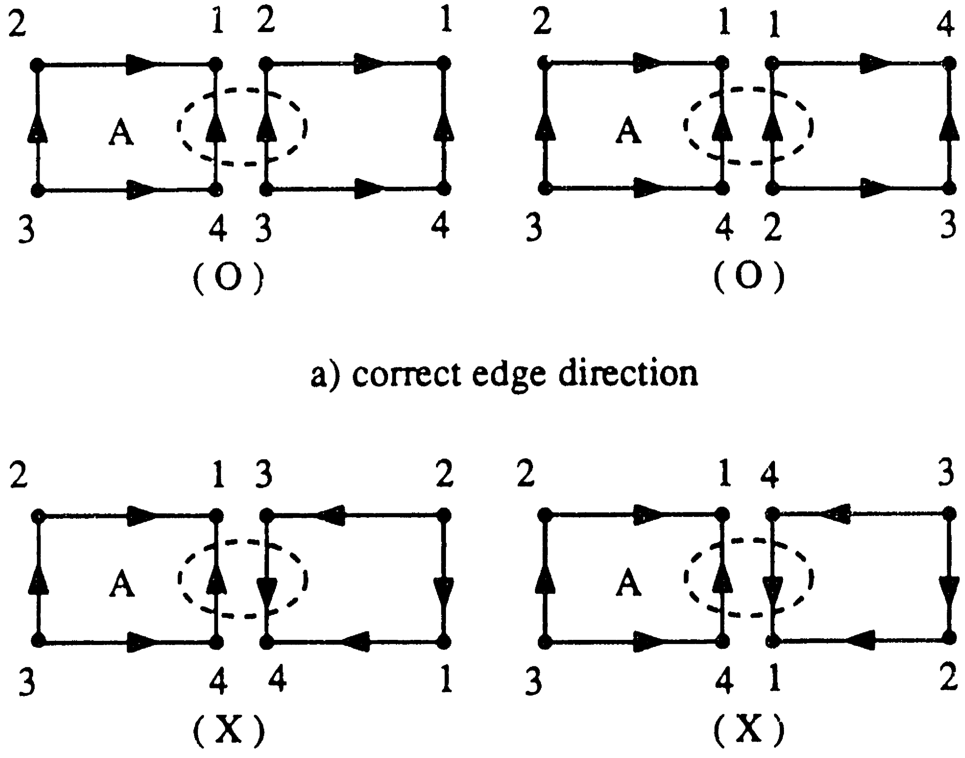


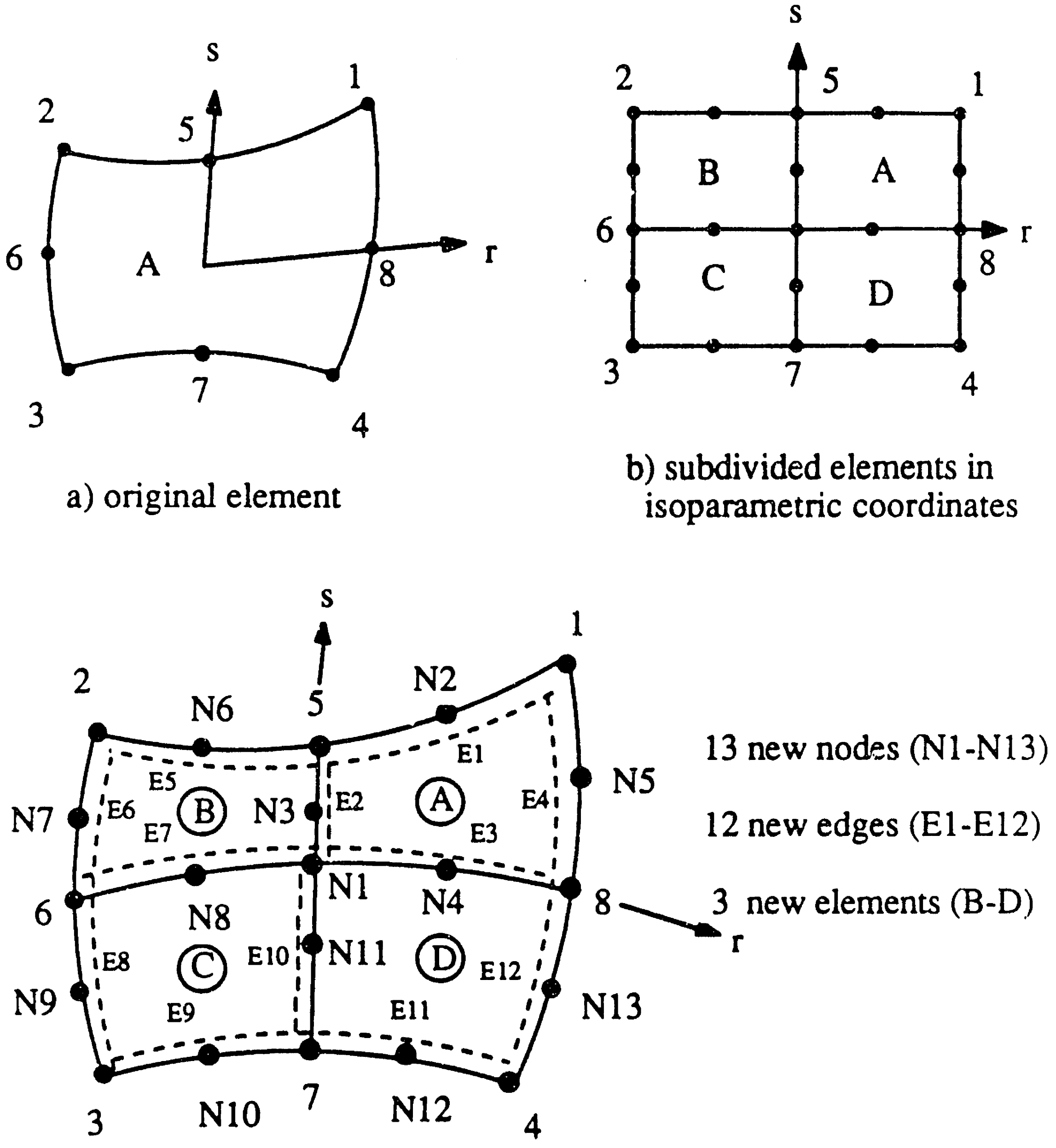
Figure 2-30: Edge numbering scheme



a) correct edge direction

a) incorrect edge direction

Figure 2-31: Examples of edge direction assignment



a) original element

b) subdivided elements in isoparametric coordinates

13 new nodes (N1-N13)

12 new edges (E1-E12)

3 new elements (B-D)

c) subdivided elements in real coordinates

Figure 2-32: Basic refinement unit for a 8-node quadrilateral element

Chapter 3

Mesh Generation for Three-Dimensional Problems

3.1 Overall Procedures for Near-Optimal Mesh Generation

A new mesh generation scheme for three-dimensional problems has been developed in this research. This scheme is applying the concept of the two-dimensional triangulation process to three-dimensional problems. The basic strategy of the volume triangulation process is as follows: After the surface triangulation process has been completed on the loop-faces of a component, tetrahedral elements are generated from the outside boundary surface toward the inside by "cutting" the corner edges of the loop-boundary edges. Since the surface triangulation is performed using the key nodes on the loop-edges, the desired local mesh density in the three-dimensional analysis domain can be easily controlled by the key node placement on the loop-edges, which is a major advantage of our scheme. In order to construct well-conditioned tetrahedral elements, it is recommended that a complex analysis domain be subdivided into several near convex subdomains. Both curved-sided elements and straight-sided elements are used for mesh generation and the finite element analysis. For these elements, the mid-side nodes are located at the middle of an edge. Since the error analysis in three-dimensional problems can be simplified by using tetrahedral elements with straight edges, only tetrahedral elements with straight edges and their mid-side nodes located at the middle of an edge are used in this work.

The schematic diagram for a near-optimal mesh generation in three-dimensional analysis is shown in Figure 3-1. As shown in Figure 3-1, a computer

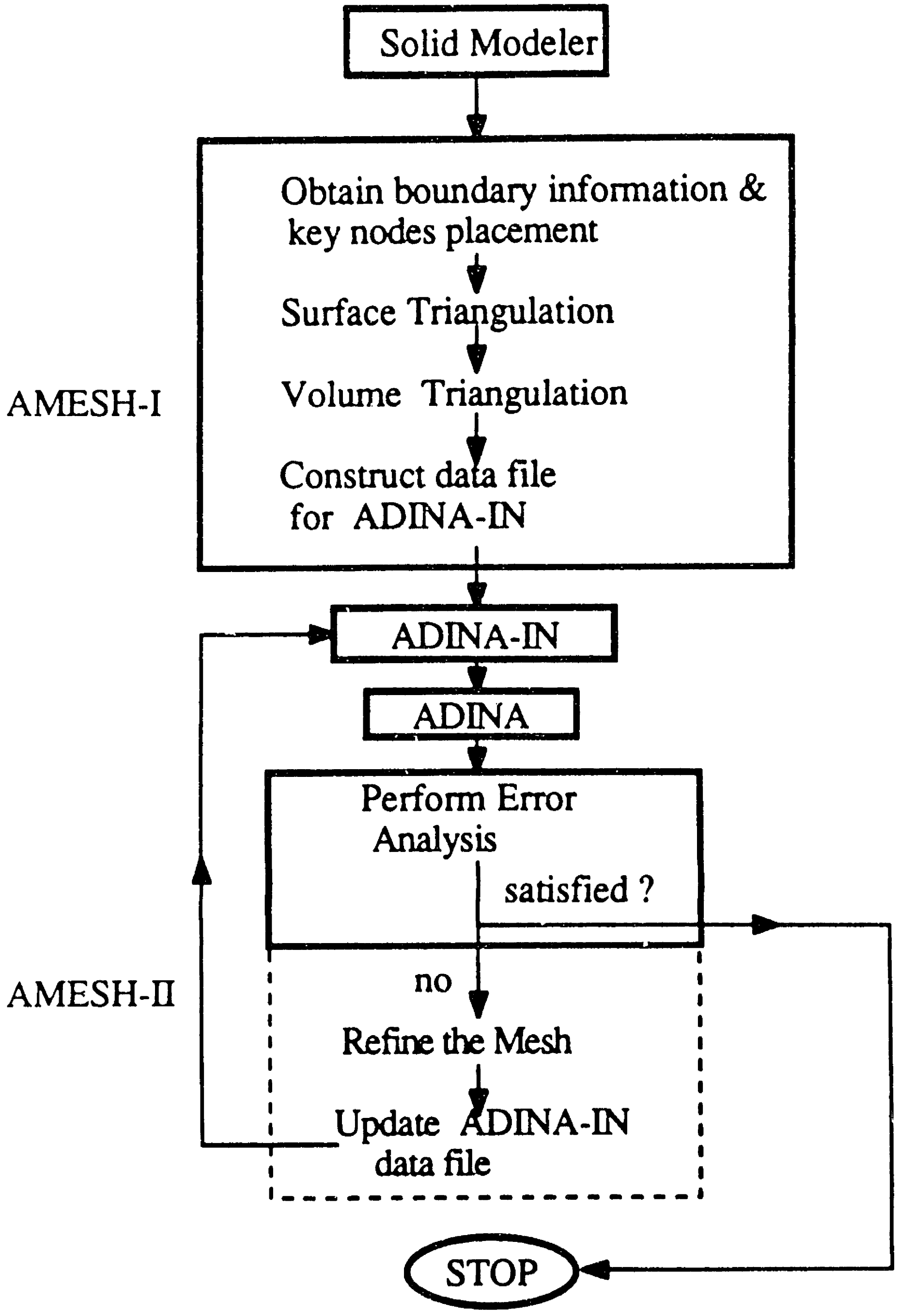


Figure 3-1: Near-optimal mesh generation process for 3-D problems

program AMESH which is composed of AMESH-I and AMESH-II has been developed. AMESH-I is designed to construct the initial mesh from a solid modeler and AMESH-II is designed to perform the error analysis. An adaptive refinement process is not included in AMESH-II yet, however it should be included to obtain a near-optimal mesh in the future.

The data for AMESH-I is of boundary representation type and it will be necessary to use an interfacing program to prepare the input data for AMESH-I if an external solid modeler (either CSG or B-REP type) is to be linked to AMESH-I directly. Same examples of the surface triangulation and volume triangulation process are shown in Figure 3-2 .

3.2 Design of the Winged-Edge/Face Data Structure

An analysis domain in the three-dimensional space can be considered to be an assembly of subcomponents, which refer to subdomains obtained by dividing the entire domain into several regions. In order to construct well-conditioned tetrahedral elements, near convex subdomains are recommended for the complete representation of the analysis domain.

A component is called a "*loop*" and is defined by its outside boundaries, i.e. loop-faces and loop-edges. A hole surface inside the analysis domain is also considered to be an outside boundary and an inside boundary does not exist in our discussion. The hierarchical representation of the data structure used is shown in Figure 3-3. In Figure 3-3, LP is a component loop enclosed by loop-faces and LF is a loop-face enclosed by loop-edges in a counter-clockwise direction. The triangulation process is to subdivide a loop-face into triangles which become real faces of tetrahedral elements. LE is a loop-edge which is an

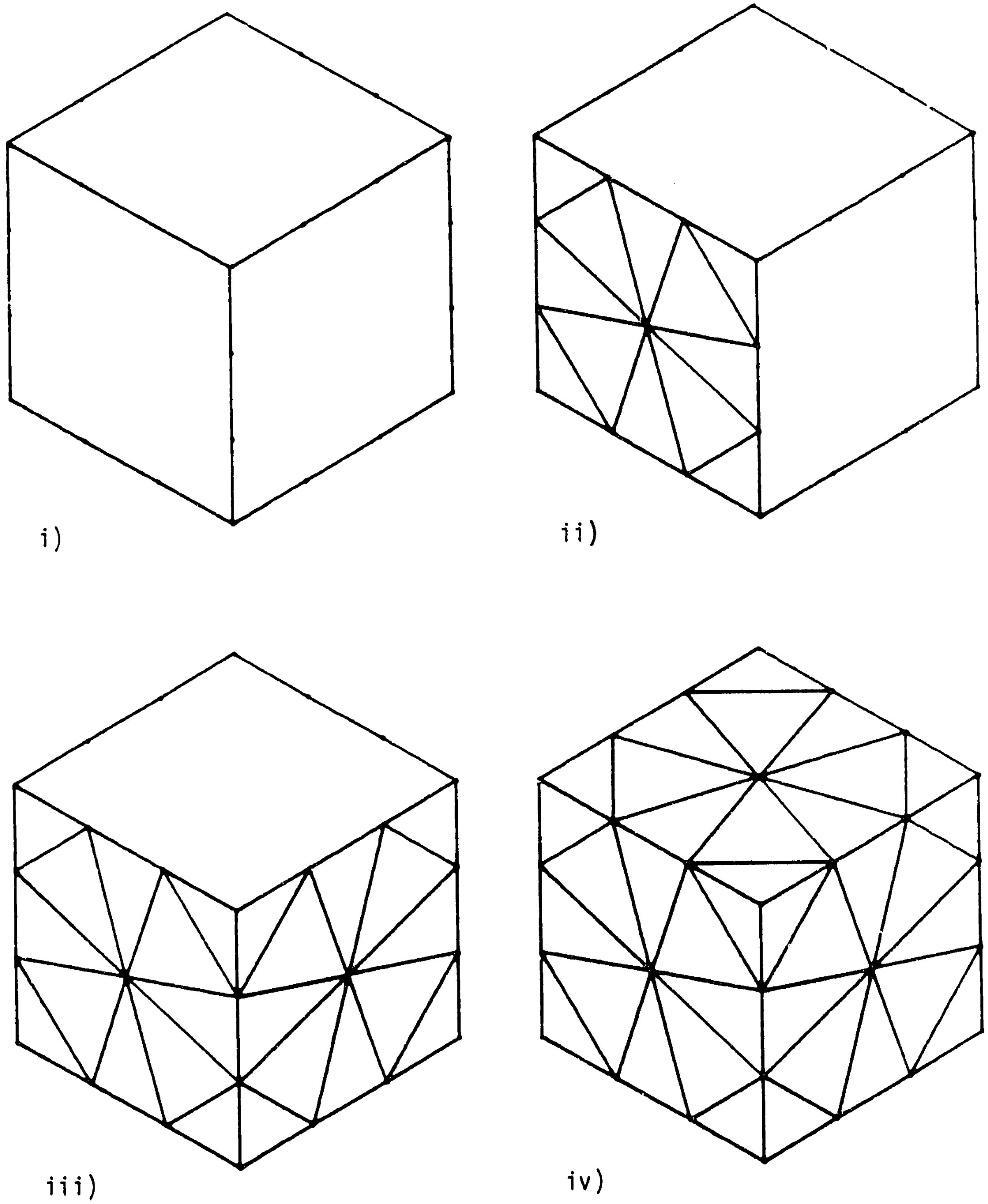


Figure 3-2: An example of the surface and volume triangulation process

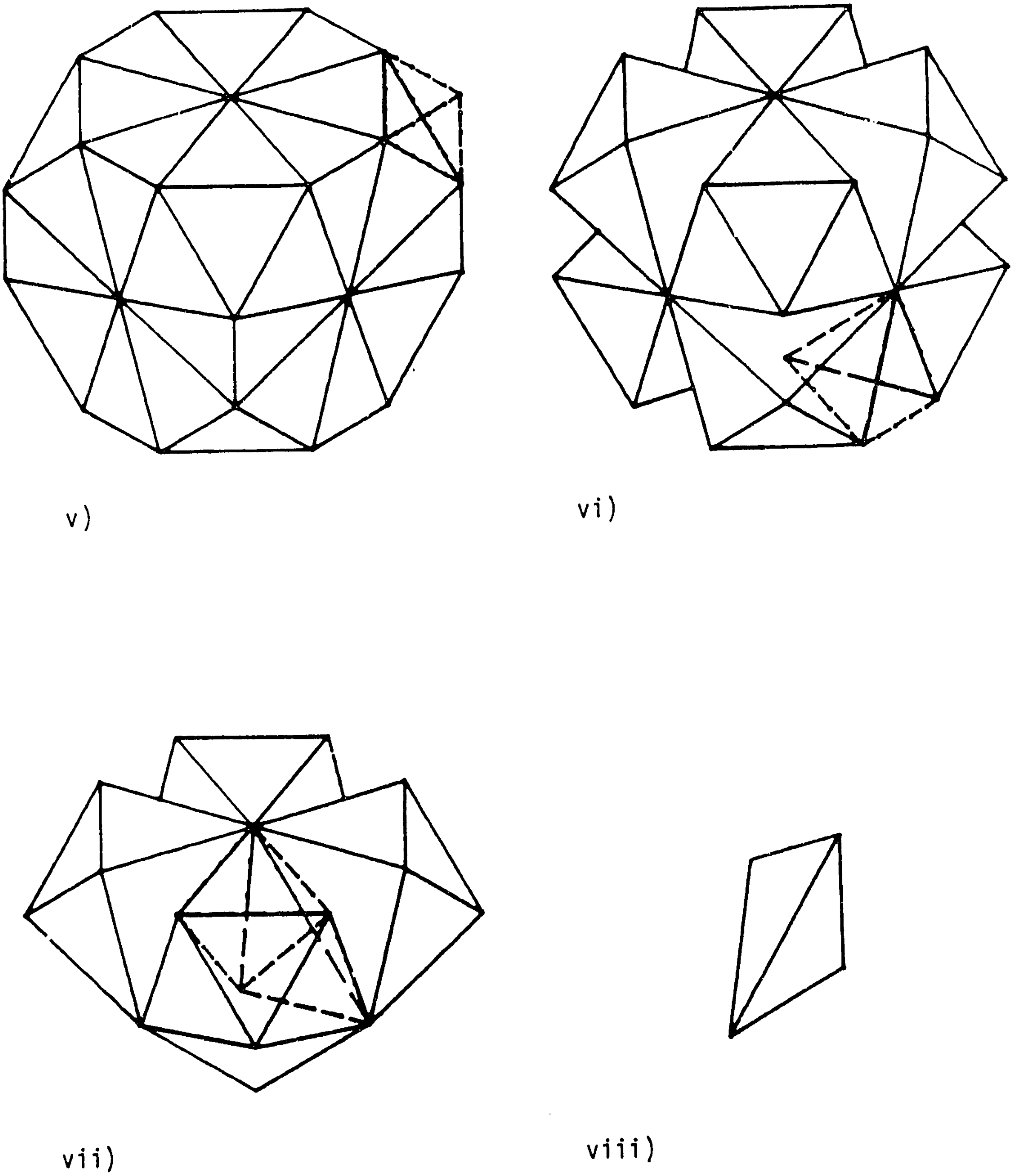


Figure 3-2: An example of the surface and volume triangulation process

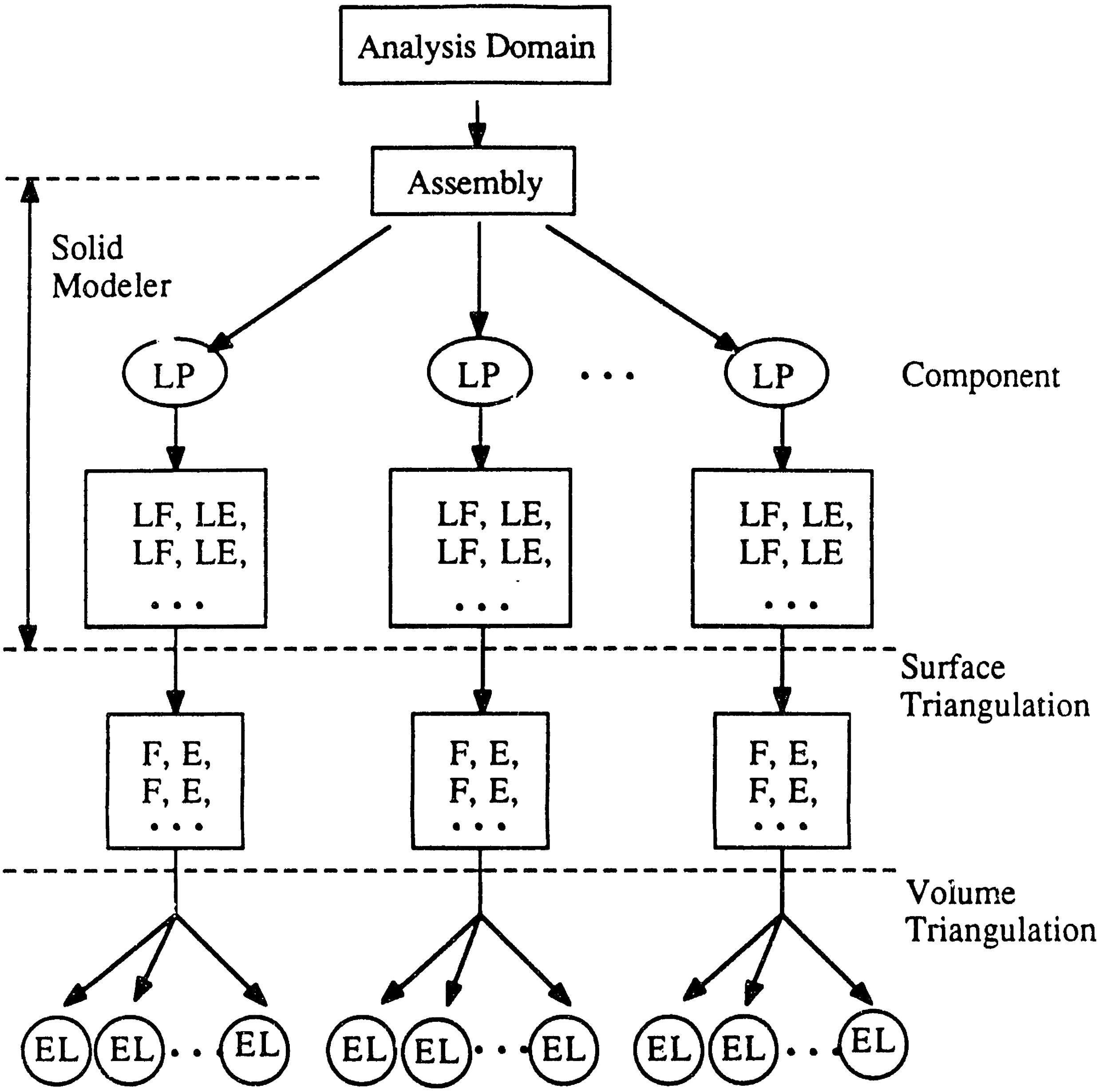


Figure 3-3: Hierarchical representation of the data structure

intersection of two or more loop-faces and is to be subdivided into edges which become real edges of tetrahedral elements. EL is a tetrahedral element, F is a real face of an element and E is a real edge of an element.

After the completion of the surface triangulation process, a component loop is described by real faces and edges instead of loop-faces and loop-edges, so the connectivity information between faces and edges on a component loop is ready for the volume triangulation process.

Since loop-edges are assigned in a counter-clockwise direction to form a loop-face, a loop-face has a certain direction vector which is normal to the loop-face and points outside of a component loop. If the direction vector points outside of a component, the component has a positive face number and if it points inside to a component, a negative face number is assigned to the component. Therefore, the boundary loop-faces of the assembly have positive face numbers and the internal loop-faces generated by subdividing the assembly into components have both positive and negative face numbers. Only positive loop-faces are triangulated and the resulting edge and face connectivity data are stored for the negative loop-faces of adjacent components. Hence, positive loop-faces should be assigned to a component which is to be triangulated first and negative loop-faces should be assigned to a component that is triangulated later. By using the positive and negative loop-faces at the interface of components, a complex analysis domain can be easily subdivided into components and be triangulated.

The whole data structure designed is composed of two parts. One part is for the solid modeler and stores the geometry information of the assembly and the other is to store the results of the surface and volume triangulation processes. The solid modeler used in our discussion is designed to store the minimum amount of data required for mesh generation and only includes planar and cylindrical surfaces. The arrays storing the geometric information and the topological information of the data structure shown in Figure 3-4 are described as follows.

User input information

LPLFACE(ncomp,i)	Loop-faces enclosing a component ex) 1, 2, -3,...
LPLEDGE(ncomp,i)	Loop-edges enclosing a component
ILFACE(nface,i)	Type of a loop-face (planar, cylindrical) Type of a cylindrical surface w.r.t. the virtual view plane (convex, concave) Number of loop-edges enclosing this loop-face Pointer to the geometric information of a cylindrical surface Type of a loop-face (normal, thin)
LFLEDGE(nface,i)	Loop-edges enclosing a loop-face ex) 1, -2, 3, ...
ILEDGE(4,nedge)	First node of a loop-edge Second node of a loop-edge Pointer to geometry (straight line, circular arc) Number of actual edges in a loop-edge
ICIR(2,ncir)	Center node number Sign of the angle between the first and second node (1,-1)
EDRATIO(nedge)	Adjacent edge length ratio in the direction of a loop-edge

Information generated in the program

UVECT(3,nface)	X-component of the unit vector of a loop-face Y-component of the unit vector of a loop-face Z-component of the unit vector of a loop-face

CYLIND(2,ncyl)	Radius of a cylidrical surface Angle of a cylindrical surface
LOOPNODE(nface)	Boundary key node enclosing a current loopboundary during the surface triangulation process
LOOPEDGE(ncomp,i)	Actual edges enclosing a component
LOOPFACE(ncomp,i)	Actual faces enclosing a component
LPOINT(i)	Flag whether a node is on a current loop-boundary or not during the surface and volume triangulation process
LFLFACE(2,nface)	First actual face in a loop-face Last actual face in a loop-face
INTEDLF(2,nface)	First actual edge within a loop-face Last actual edge within a loop-face
INTNDLF(2,nface)	First key node within a loop-face Last key node within a loop-face
LESEDGE(2,nedge)	First actual edge in a loop-edge Last actual edge in a loop-edge
INTNDLE(2,nedge)	First internal node in a loop-edge Last internal node in a loop-edge
IFACE(3,nface)	First node of a face Second node of a face Third node of a face
JFACE(5,nface)	First edge of a face Second edge of a face Third edge of a face Positive side element of a face Negative side element of a face
IEDGE(5,nface)	First node of an edge Second node of an edge Third node of an edge Left face of an edge Right face of an edge

JELEM(4,nelem)	First face of an element Second face of an element Third face of an element Fourth face of an element
NOD(10,nelem)	Nodes of an element
INTNDOBJ(ncomp,2)	First key node inside a component Last key node inside a component
INTEDOBJ(ncomp,2)	First edge inside a component Last edge inside a component
INTELOBJ(ncomp,2)	First element inside a component Last element inside a component

The connectivity information in our solid modeler is shown in Figure 3-5. As shown in Figure 3-5, there is no redundant information stored and the data to be stored can be easily input by the user or extracted from the external solid modeler if used. The connectivity information after surface and volume triangulation is shown in Figure 3-6.

Although the adaptive refinement process for tetrahedral elements is not included in this research, the data structure designed can be easily extended to cover the refinement process.

3.3 Surface triangulation

Surface triangulation in three-dimensional space can be performed in three steps. First, a surface with key nodes is transformed into a *view plane* where the view direction is parallel to the surface normal, and then the triangulation process described in Chapter 2 is performed on this plane. Finally, the triangulated domain in the view plane is transformed back to the original surface of the object. The transformation of the data between a three-dimensional space

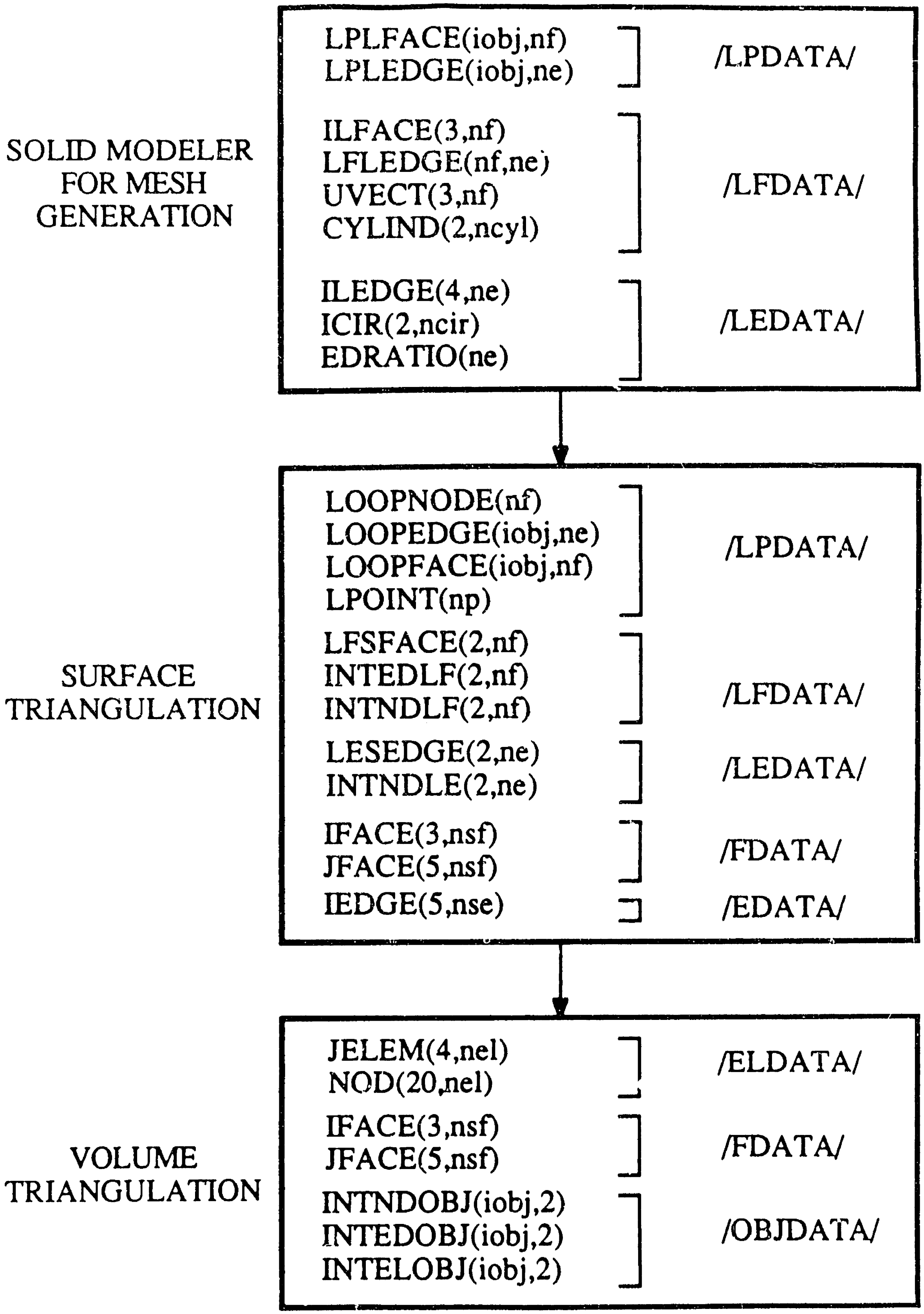


Figure 3-4: Dissembled winged-edge/face data structure

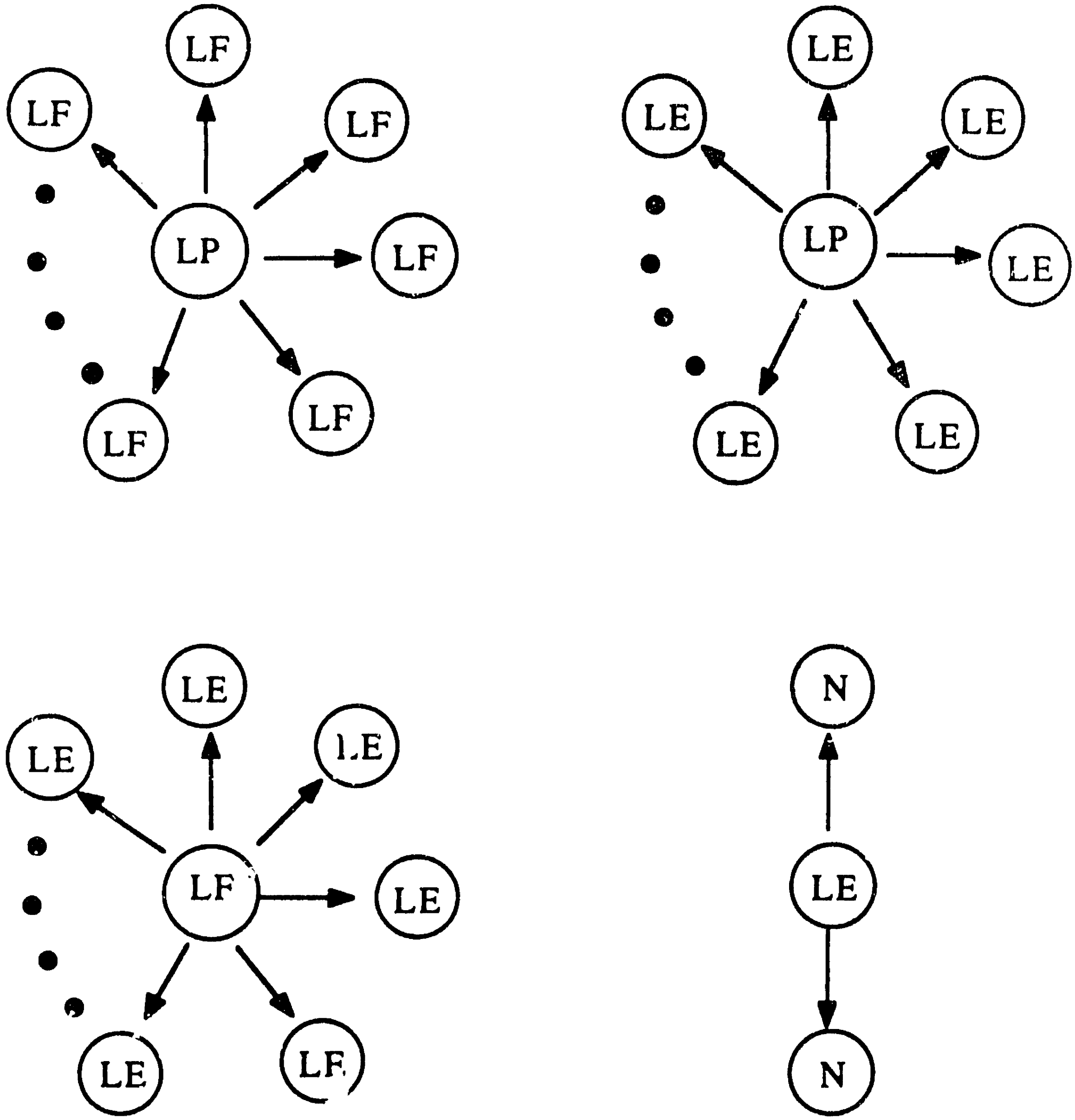


Figure 3-5: Connectivity information in the solid modeler

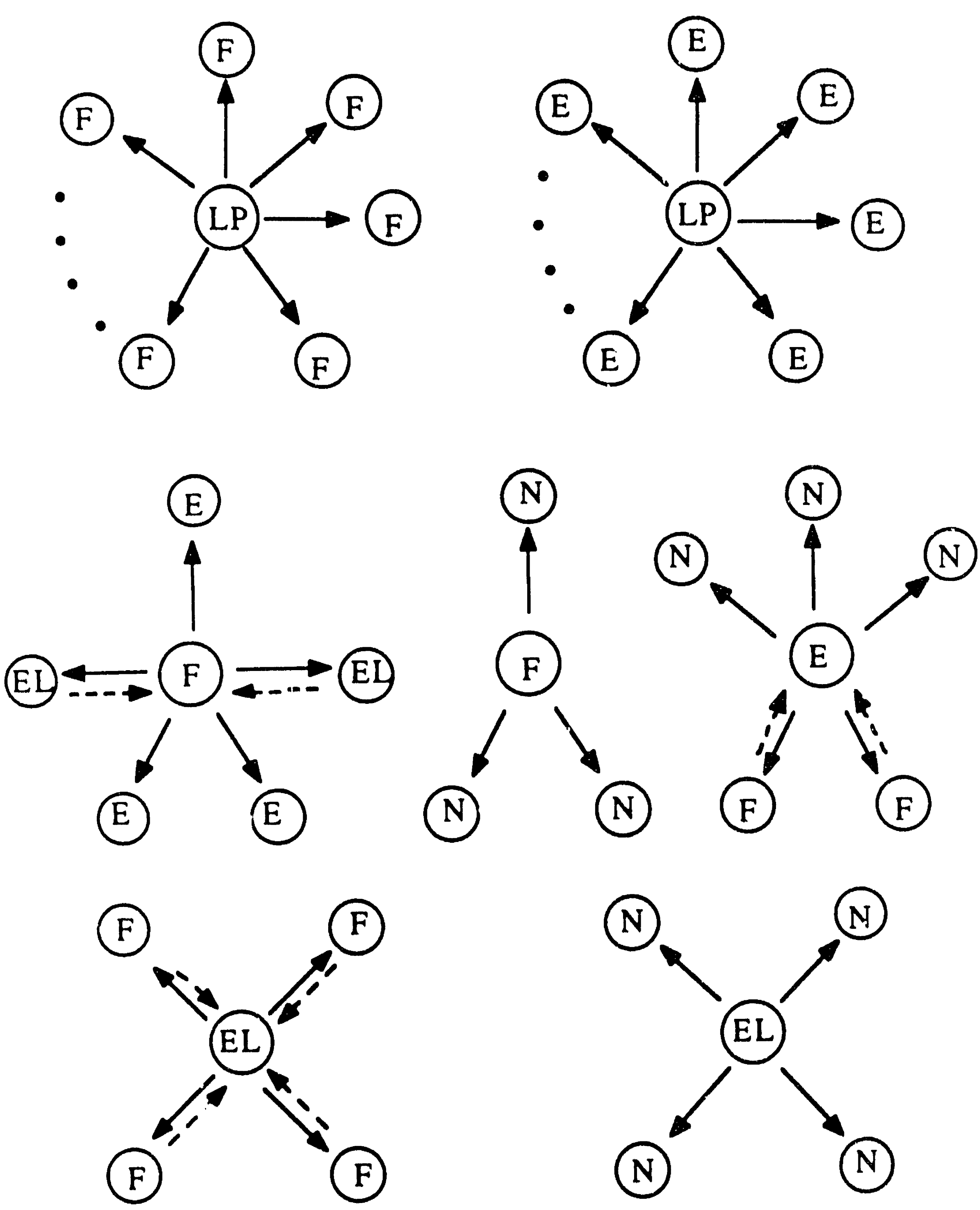


Figure 3-6: Connectivity information for the surface and volume triangulation

and a two-dimensional view plane is performed by using the parallel projection technique [55-56], in which the view direction is parallel to the surface normal and passes through the origin of the global coordinates. In our discussion, only planar and cylindrical surfaces are considered.

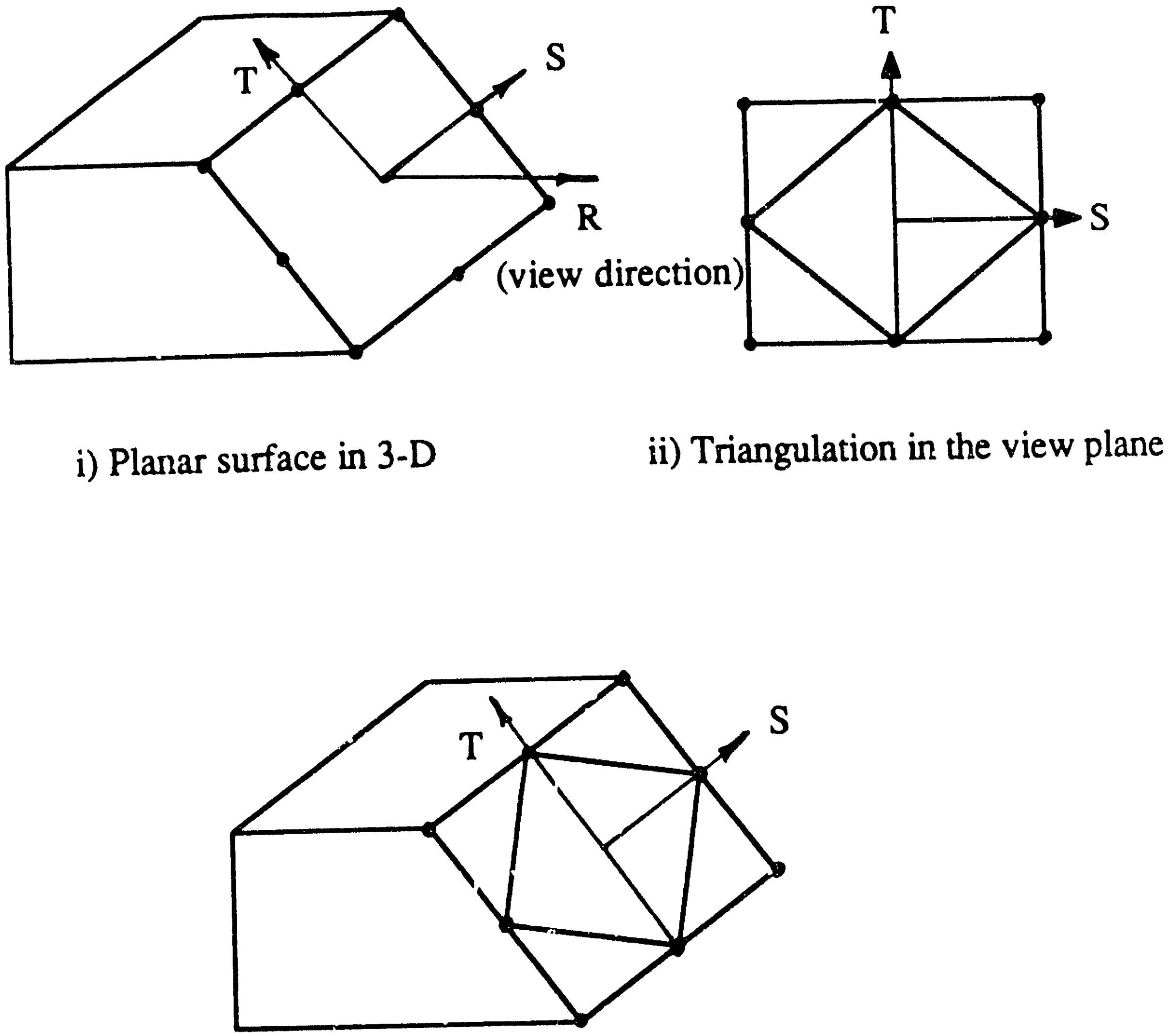
A planar surface on which the surface normal is uniquely defined can be transformed to a view plane by using the surface normal vector. The example of planar surface triangulation is shown in Figure 3-7.

In case of a cylindrical surface for which the surface normal vector is not uniquely defined, a virtual view plane and the corresponding normal vector are used for the triangulation. A cylindrical surface can be considered to be a domain enclosed by two circular loop-edges and two straight loop-edges. Here, a *virtual view plane* is defined as the one enclosed by the straight loop-edges and the chords of circular loop-edges. The surface in this view plane is scaled up by using a scale factor, s_r, in order not to reduce the actual size of the circular loop-edges as shown in Figure 3-8.

$$S_r = \frac{l_{perimeter}}{l_{chord}} \qquad (3-1)$$

So the resulting surface in the virtual view plane is equivalent to the unfolded strip of a cylindrical surface.

Once the surface triangulation process has been completed on the unfolded strip of a cylindrical surface, the resulting mesh is transformed to the cylindrical surface in the view direction as shown in Figure 3-9. Finally, the triangulated cylindrical surface in the view direction is transformed back to the original surface of the object.

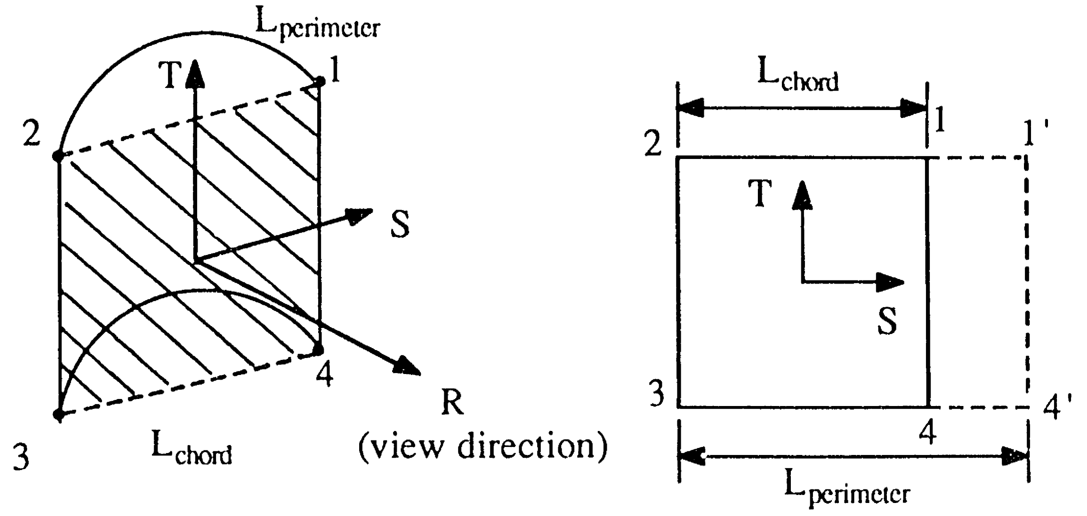


i) Planar surface in 3-D

ii) Triangulation in the view plane

iii) Back transformation to original suface

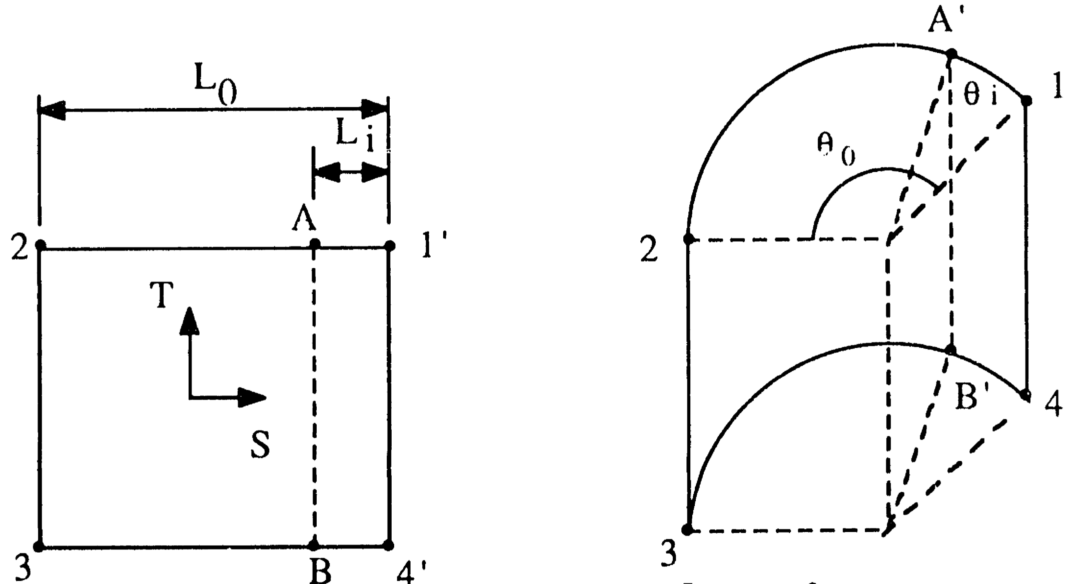
Figure 3-7: Planar surface triangulation



a) cylindrical surface (concave)

b) surface in a virtual view plane

Figure 3-8: Cylindrical surface in a virtual view plane



transformation rule ; $\frac{L_i}{L_0} \cdot = \frac{\theta_i}{\theta_0}$

a) line A-B in a virtual view plane

b) link A'-B' in cylindrical surface

Figure 3-9: Transformation to cylindrical surface in the view direction

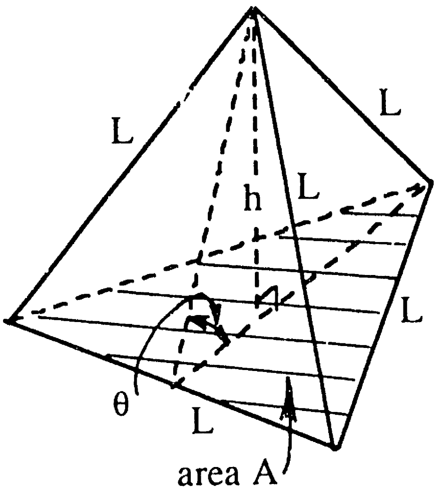
3.4 Volume triangulation

The basic idea of volume triangulation can be considered to be an extension of surface triangulation. As is the case of surface triangulation, volume triangulation starts from the outside boundary toward the inside by removing the corner edges of a component loop-boundary using certain basic operations. With the surface triangulation process, all the edges on a loop-boundary are assigned to be "level one" and as the volume triangulation proceeds, the "levels" of the new generated edges are increased by one for each operation.

Since the best form of tetrahedral elements in finite element analysis can be considered to be equilateral tetrahedra, the resulting elements are generated as close as possible to equilateral tetrahedra. The desired tetrahedron is shown in Figure 3-10.

Due to the topological requirements, which will be described later in this chapter, at least three basic operations are needed to reduce the number of edges and faces in a loop-boundary and in our scheme, one more basic operation is designed to construct the last two or three tetrahedral elements. In order to describe the basic operations, the following definitions are introduced. Consider an edge, KE, in Figure 3-11.

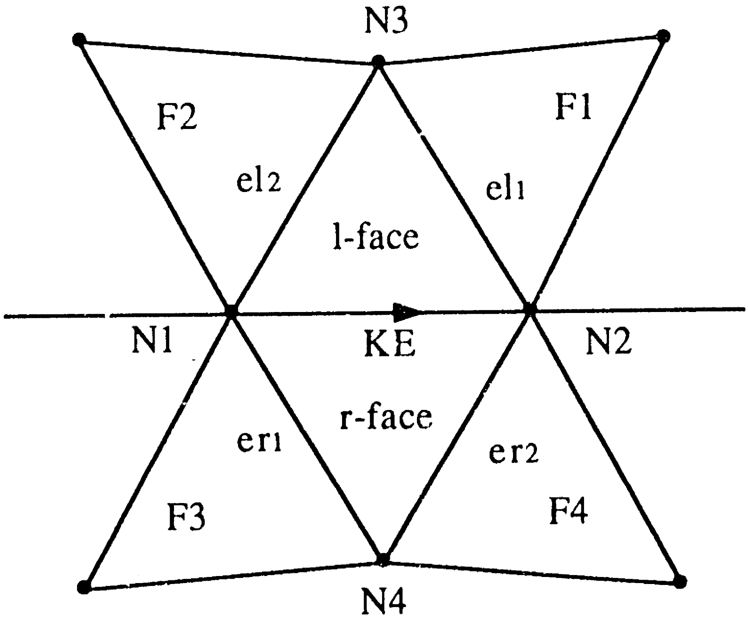
An edge KE has two adjacent faces, a left face and a right face, and four *surrounding edges*, el1,el2,er1 and er2, in a counter-clockwise direction on each side of the edge. The adjacent faces of these surrounding edges, el1,el2,er1,er2, are F1,F2,F3 and F4 respectively and are called *surrounding faces* to an edge KE. An *edge angle* is a dihedral angle formed by two adjacent faces, the left face and the right face.



$\theta \approx 70.53^{\circ}$

$h \approx 0.8165\,L \approx 1.24\sqrt{A}$

Figure 3-10: Desired tetrahedral element



surrounding edges ;

el1 el2 er1 er2

surrounding faces ;

F1 F2 F3 F4

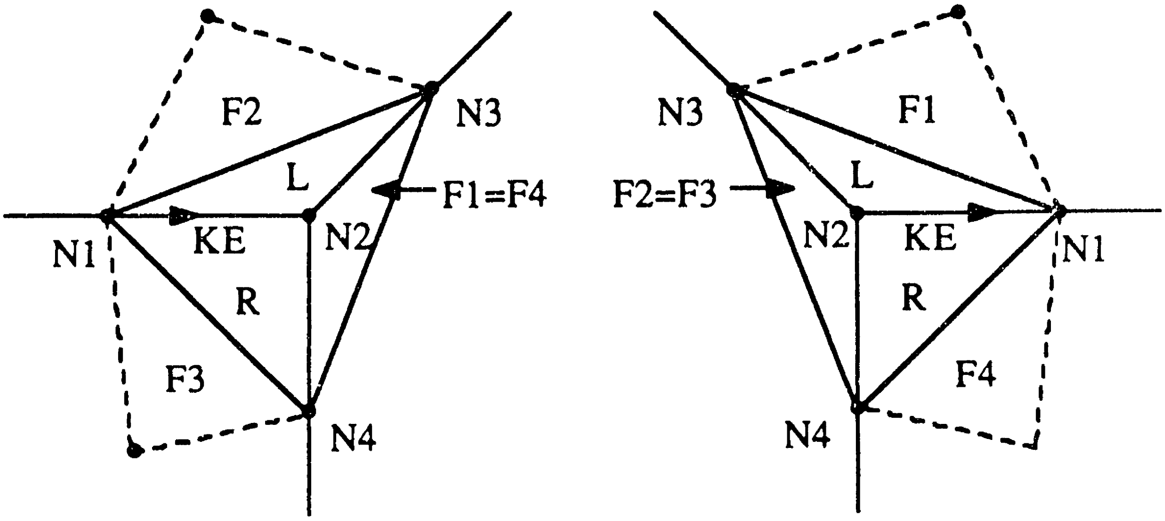
Figure 3-11: Unfolded view around edge KE

Four basic operations (Type-1A,Type-1B,Type-2,Type-0) are designed to perform the volume triangulation. Type-1A and Type-1B operations are designed to generate one element at a time from a loop-boundary and Type-2 operation is designed to generate two elements. Type-0 operation is designed to construct the last two or three elements in order to complete the volume triangulation process.

Topologically, a Type-1A edge is an edge which has one common surrounding face as shown in Figure 3-12. This is equivalent to the condition that only three edges meet at one node. However, in order to generate well-conditioned elements, geometric restrictions are imposed ,i.e. the edge angle should be less than or equal to 120°. A Type-1B edge is an edge for which the edge angle is less than 85° ($\delta_i \geq 1.8$) or 100° ($\delta_i < 1.8$) depending on the face area ratio, δ_i, and certain conditions are satisfied. Since, the desired element has an edge angle of 70°,a Type-2 edge where two elements are constructed at a time can have a maximum edge angle of 175°.

Type-1A operation

A Type-1A operation on a Type-1A edge generates one tetrahedron by removing three faces, three edges and one key node from a loop-boundary and generates one new face as shown in Figure 3-13. This operation can be considered to be *trimming* process. In this operation, the derivative of Euler's formula is satisfied as follows.



a) F1=F4 or

3 edges meet at node N2

a) F2=F3 or

3 edges meet at node N1

Figure 3-12: Type-1A edge

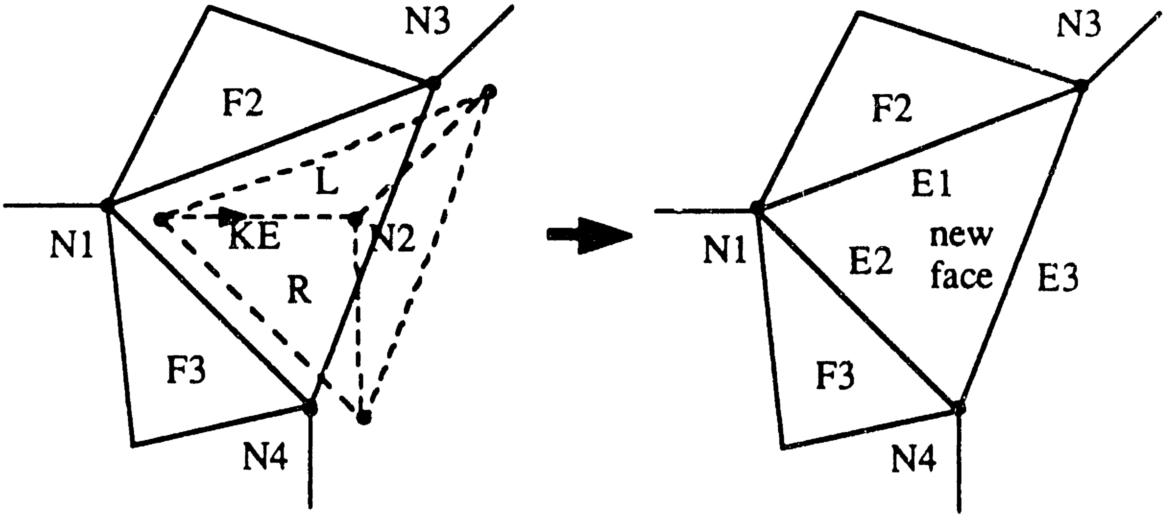


Figure 3-13: Type-1A operation - trimming

$$\Delta V - \Delta E + \Delta F = -1 + 3 - 2 = 0$$

$$\Delta V = -1 \ ; \ change \ in \ number \ of \ key \ nodes$$

$$\Delta E = -3 \ ; \ change \ in \ number \ of \ edges$$

$$\Delta F = -2 \ ; \ change \ in \ number \ of \ faces$$

After a Type-1A operation, the number of loop-boundary edges and faces has been reduced (ΔE=-3, ΔV=-1, ΔF=-2), and the edge angle of the remaning edges E1, E2, E3 have been reduced. In addition, the topological conditions on nodes N1, N2, N3 have also been changed so that the number of edges connected to these nodes have been reduced by one. Therefore, this operation can generate new Type-1A edge around nodes N1, N2, N3, when the number of edges connected to these nodes reaches three due to this operation.

Type-1B operation

This operation is designed to generate one tetrahedral element at a Type-1B corner edge as shown in Figure 3-14. This operation can be considered to be *wedging* process. In this operation, one edge and two faces have been removed, and instead one new edge and two new faces have been introduced. Therefore, the derivative of Euler's formula is also satisfied as follows.

$$\Delta V - \Delta E + \Delta F = 0$$

$$(\Delta V = 0, \quad \Delta E = 0, \quad \Delta F = 0)$$

The changes due to this operation are as follows. Geometrically, the edge angles of surrounding edges el1, el2, er1, er2 have been decreased so that it promotes to

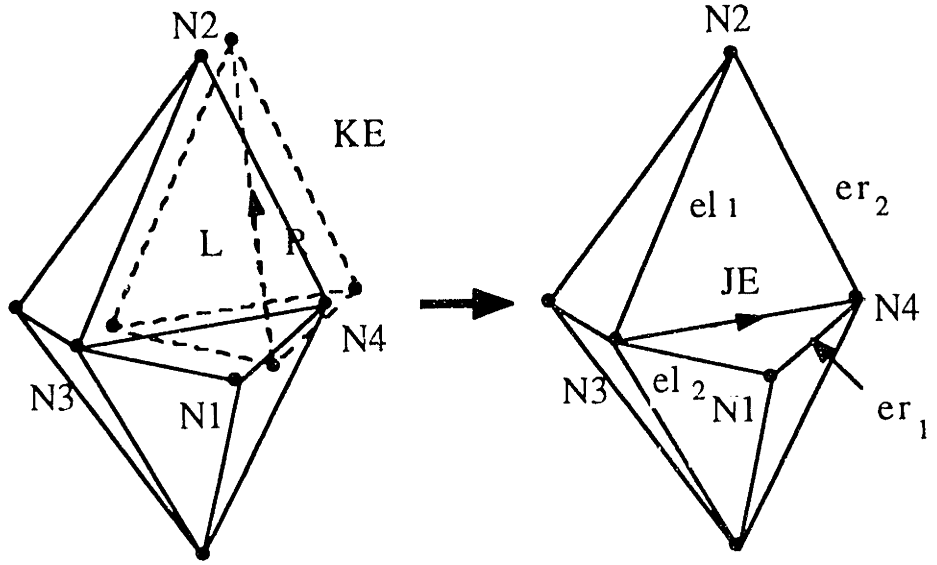


Figure 3-14: Type-1B operation - wedging

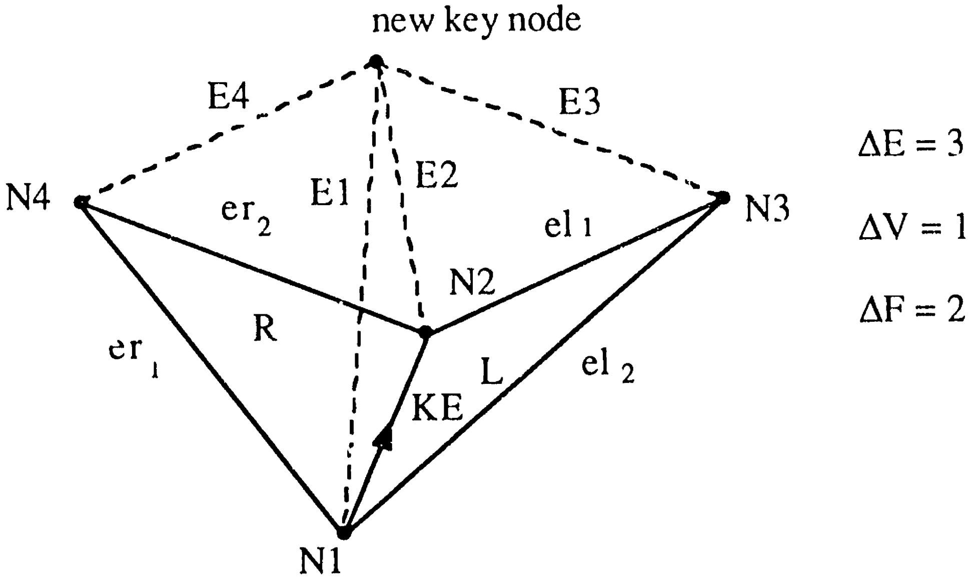


Figure 3-15: Type-2 operatiun - digging

generate Type-1A, Type-1B or Type-2 edges. Topologically, for nodes N1 and N2, the number of edges connected to these nodes has decreased by one due to the removal of an edge KE, while for nodes N3 and N4, the number of edges has increased by one due to the introduction of new edge JE. Hence, this operation can generate new Type-1A edges around nodes N3 and N4.

Type-2 operation

This operation is designed to generate two tetrahedral elements at a Type-2 corner edge by introducing a new key node as shown in Figure 3-15. This operation can be considered to be *digging* process.

In this operation, an edge of interest KE is removed and instead four new edges E1, E2, E3, E4 have been introduced to a loop-boundary , so the change in the number of edges, ΔE is +3. Also two adjacent faces, left and right face, are removed and four new faces are generated, thus the change in the number of faces, ΔF is +2. Since a new key node is introduced, the number of key nodes on a loop-boundary is increased by one. Thus, this operation satisfies the derivative of Euler's formula as follows.

$$\Delta V - \Delta E + \Delta F = 1 - 3 + 2 = 0$$

The geometrical contribution of this operation to the entire volume triangulation process is to reduce the surrounding edge angles at el1, el2, er1, er2 so that it promotes to generate Type-1A, Type-1B and Type-2 edges among the edges el1,el2,er1 and er2. Topologically, the number of edges connected to nodes N3, N4 is increased by one, while the number of edges connected to nodes N1, N2 remains unchanged. Thus, the primary goal of this operation is to generate

Type-1B edges and Type-2 edges by reducing the edge angles. However, sometimes this operation can generate Type-1A edges by reducing the edge angle when an edge has one common surrounding faces and the edge angle is greater than 120°.

Since the number of edges, faces and nodes of the loop-boundary is increased by this operation, this operation has adverse effects on the volume triangulation process, in which the number of edges , faces and nodes is to be reduced. However, this operation is sometimes necessary to generate Type-1A or Type-1B edges when no more Type-1A and Type-1B operation is possible.

A Type-2 operation, in which a new key node is generated, is a more complicated one than Type-1A and Type-1B operation, in which only the connectivity information of a loop-boundary need to be changed. A Type-2 operation is designed to be *adaptive* and is composed of two steps. First a new key node is generated considering the effects of surrounding faces and edges and then, the new key node position is adjusted according to the scaling factor, *r*. Consider the first step of a Type-2 operation in Figure 3-16. In order to construct two well-conditioned tetrahedral elements, the height of a new face, l_{12} is taken as $\sqrt{l_1 l_2}$, the geometric mean of those of adjacent faces with the following conditions,

$$\frac{l_1}{l_{12}} = \frac{l_{12}}{l_2} \qquad or \qquad l_{12} = \sqrt{l_1 l_2} \qquad\qquad (3\text{-}2)$$

Using the height of a new face, the new key node position A_1 is determined as the average of the positions, A_a and A_b.

$$OA_1 = \frac{1}{2}(OA_a + OA_b) \qquad\qquad (3\text{-}3)$$

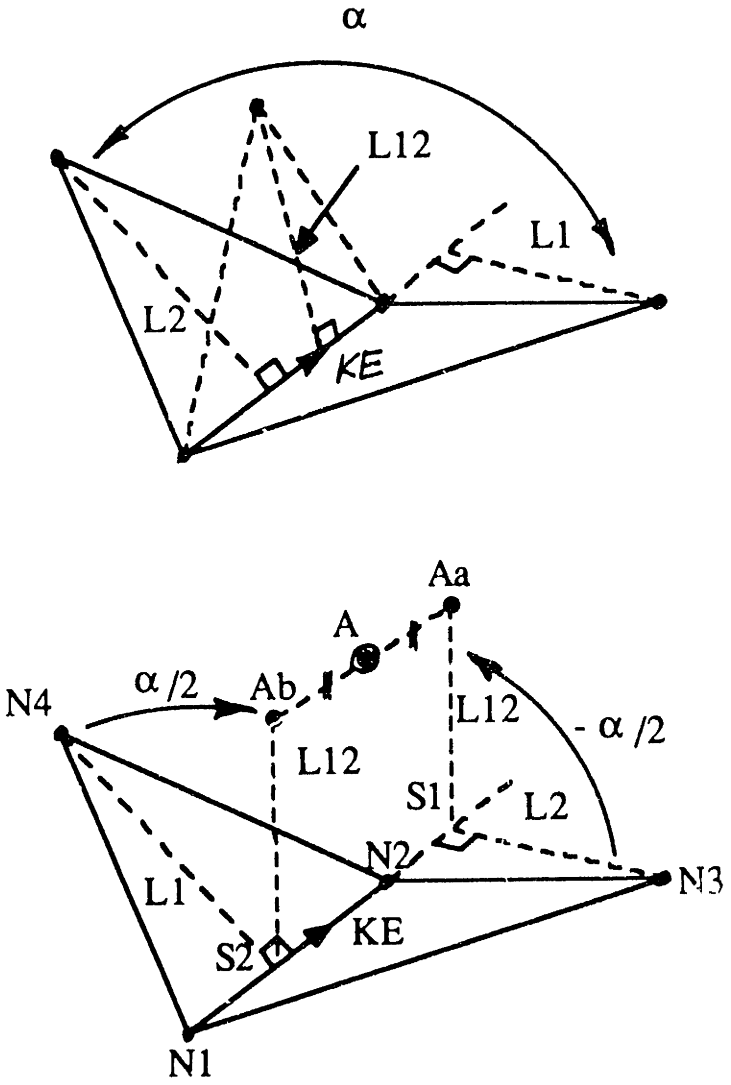


Figure 3-16: Generation of a new key node

The position A_a is dertermined by rotating a vector of length l_{12} in the direction S1-N3 with respect to the axis N1-N2 by the angle of -α/2[55].

$$\overrightarrow{OS_1} = \overrightarrow{ON_1} + \overrightarrow{N_1S_1} \qquad (3\text{–}4a)$$

$$= \overrightarrow{ON_1} + (\overrightarrow{N_1N_3} \cdot \vec{e}_1)\vec{e}_1$$

$$\textit{where} \qquad \vec{e}_1 = \frac{\overrightarrow{N_1N_2}}{|N_1N_2|}$$

$$\overrightarrow{OA_a} = \overrightarrow{OS_1} + \overrightarrow{S_1A_a} \qquad (3\text{–}4b)$$

$$= \overrightarrow{OS_1} + l_{12} \cdot [\cos(-\frac{\alpha}{2}) \cdot \vec{e}_2 + \sin(-\frac{\alpha}{2}) \cdot \vec{e}_1 \times \vec{e}_2]$$

$$\textit{where} \qquad \vec{e}_2 = \frac{\overrightarrow{S_1N_3}}{|S_1N_3|}$$

Similarly, the position A_b is determined by rotating a vector of length l_{12} in the direction S2-N4 with respect to the axis N1-N2 by the angle of α/2 as follows.

$$\overrightarrow{OS_2} = \overrightarrow{ON_1} + \overrightarrow{N_1S_2} \qquad (3\text{–}5a)$$

$$= \overrightarrow{ON_1} + (\overrightarrow{N_1N_4} \cdot \vec{e}_1)\vec{e}_1$$

$$\overrightarrow{OA_b} = \overrightarrow{OS_2} + \overrightarrow{S_2A_b} \qquad (3\text{–}5b)$$

$$= \overrightarrow{OS_2} + l_{12} \cdot [\cos(\frac{\alpha}{2}) \cdot \vec{e}_3 + \sin(\frac{\alpha}{2}) \cdot \vec{e}_1 \times \vec{e}_3]$$

$$\textit{where} \qquad \vec{e}_3 = \frac{\overrightarrow{S_2N_4}}{|S_2N_4|}$$

In order to include the effects of surrounding edges and faces, the same procedure is repeated at the surrounding edges, el1, el2, er1 anc' er2, to obtain the candidate positions, A_2, A_3, A_4 and A_5 respectively.

The number of tetrahedral elements that can be generated at an edge "i" with an edge angle ϕ_i is taken as the nearest integer of $\phi_i/70°$ so that the tetrahedra are as equilateral as possible. In our scheme, if a surrounding edge angle is greater than 245° when more than three elements should be generated, the effect of this edge is neglected for convenience. If the surrounding edge angle is less than or equal to 175° when two elements are generated, the new key node is generated using the same procedures as above. If the surrounding edge angle is between 175° and 245°, the new key node is generated as shown in Figure 3-17. In order to construct three well-conditioned elements, the height of a new face l_i is taken as

$$l_i = \sqrt[3]{l_a{}^2 l_b} \tag{3–6}$$

The new key node position A_i is obtained as the weighted average of the positions A_a and A_b such as

$$\overrightarrow{OA_i} = \frac{1}{3}(2 \cdot \overrightarrow{OA_a} + \overrightarrow{OA_b}) \tag{3–7}$$

The position of A_a is determined as,

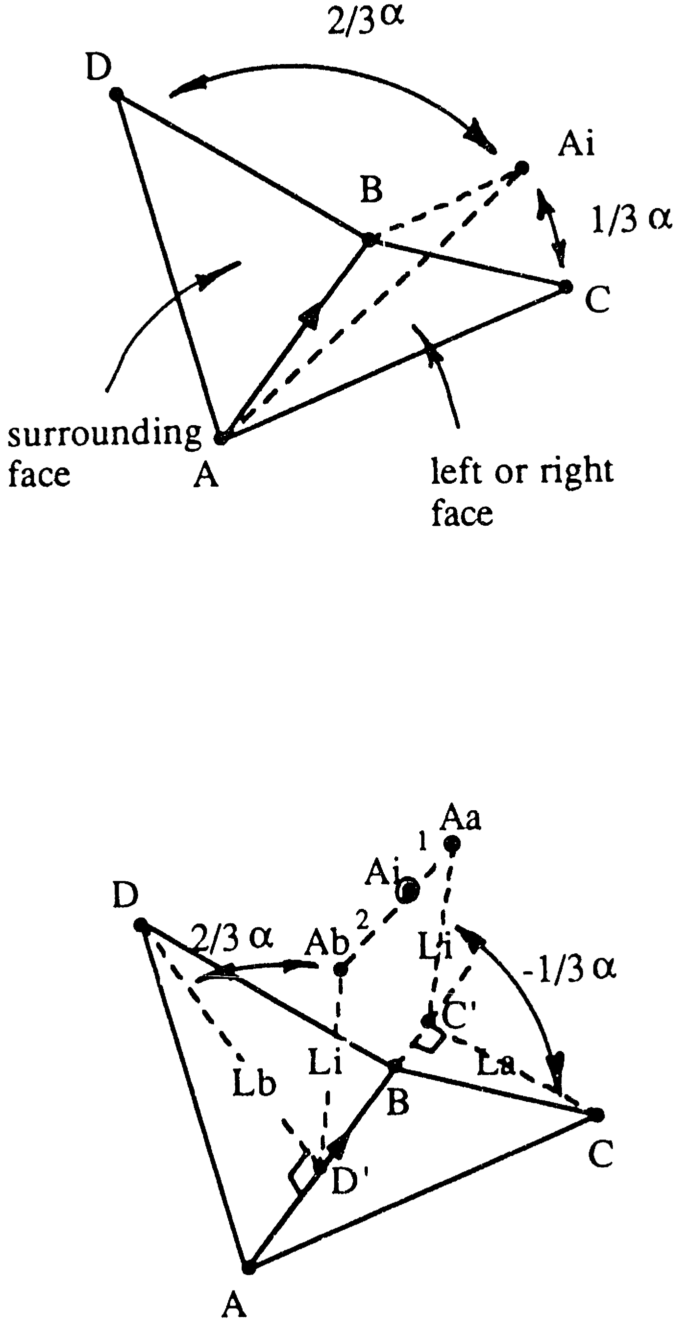


Figure 3-17: New key node position from a surrounding edge

$$\overrightarrow{OC'} = \overrightarrow{OA} + \overrightarrow{AC'} \tag{3-8a}$$

$$= \overrightarrow{OA} + (\overrightarrow{AC} \cdot \vec{e}_1)\vec{e}_1$$

where $\vec{e}_1 = \dfrac{\overrightarrow{AB}}{|AB|}$

$$\overrightarrow{OA_a} = \overrightarrow{OC'} + \overrightarrow{C'A_a} \tag{3-8b}$$

$$= \overrightarrow{OC'} + l_i \cdot [\cos(-\frac{\alpha}{3}) \cdot \vec{e}_2 + \sin(-\frac{\alpha}{3}) \cdot \vec{e}_1 \times \vec{e}_2]$$

where $\vec{e}_2 = \dfrac{\overrightarrow{C'C}}{|C'C|}$

Similarly, the position of A_b is determined as,

$$\overrightarrow{OD'} = \overrightarrow{OA} + \overrightarrow{AD'} = \overrightarrow{OA} + (\overrightarrow{AD} \cdot \vec{e}_1)\vec{e}_1 \tag{3-9a}$$

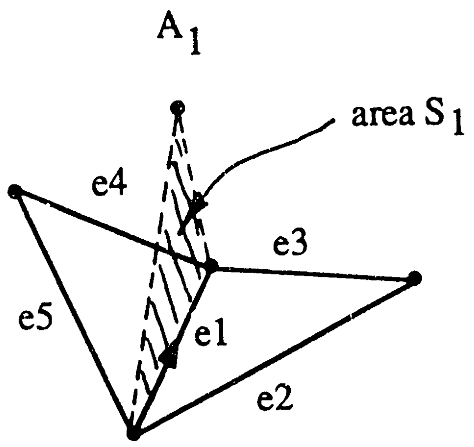
$$\overrightarrow{OA_b} = \overrightarrow{OD'} + \overrightarrow{D'A_b} \tag{3-9b}$$

$$= \overrightarrow{OD'} + l_i \cdot [\cos(\frac{2\alpha}{3}) \cdot \vec{e}_3 + \sin(\frac{2\alpha}{3}) \cdot \vec{e}_1 \times \vec{e}_3]$$

where $\vec{e}_3 = \dfrac{\overrightarrow{D'D}}{|D'D|}$

After obtaining five positions for a new key node from an edge of interest and four surrounding edges, the weighting factors are used to determine a unique position of a new key node. Since the shape of small elements will be more distorted by using a modified new node, the weighting factors are taken in inverse proportions to the area of each new face as shown in Figure 3-18.

After obtaining the new key node position A, the next step of an *adaptive* Type-2 operation is to adjust the key node position from A to A^* using a scaling factor r, as shown in Figure 3-19. The scaling factor, r, is defined as



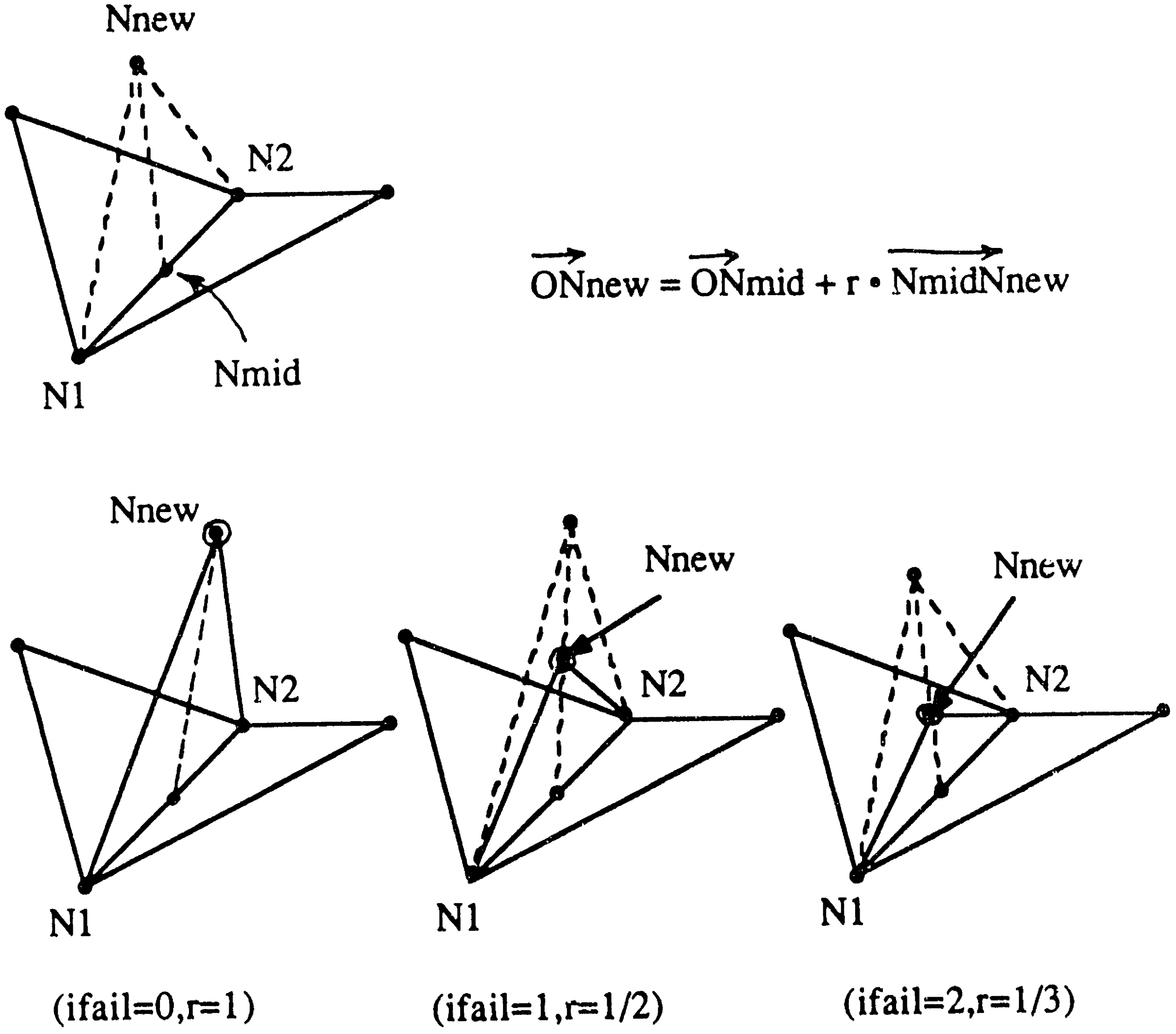
$$w'_i \propto \frac{1}{S_i} \quad , \quad i = 1,\ 2, \cdots ,\ 5$$

$$Let \qquad w'_i = \frac{1}{S_i} \cdot \frac{1}{w'_0}$$

$$where \quad w'_0 = \sum_{i=1}^{5} \frac{1}{S_i} = \left(\frac{1}{S_1} + \frac{1}{S_2} + \cdots + \frac{1}{S_5}\right)$$

$$OA = w'_1 \cdot OA_1 + w'_2 \cdot OA_2 + w'_3 \cdot OA_3 + w'_4 \cdot OA_4 + w'_5 \cdot OA_5$$

Figure 3-18: Key node position using the weighting factors



$$\overrightarrow{ONnew} = \overrightarrow{ONmid} + r \bullet \overrightarrow{NmidNnew}$$

(ifail=0,r=1) (ifail=1,r=1/2) (ifail=2,r=1/3)

Figure 3-19: Adjustment of a new key node position

$$r = \frac{1}{1 + ifail} \quad , \qquad ifail = 0 , 1 , 2 ,... \qquad (3\text{--}10)$$

where *ifail* is a number of failure in performing any basic operation. Initially, *ifail* is set to zero and if no more basic operation is possible because of failure in check process, *ifail* is increased by one and so on.

Type-0 operation

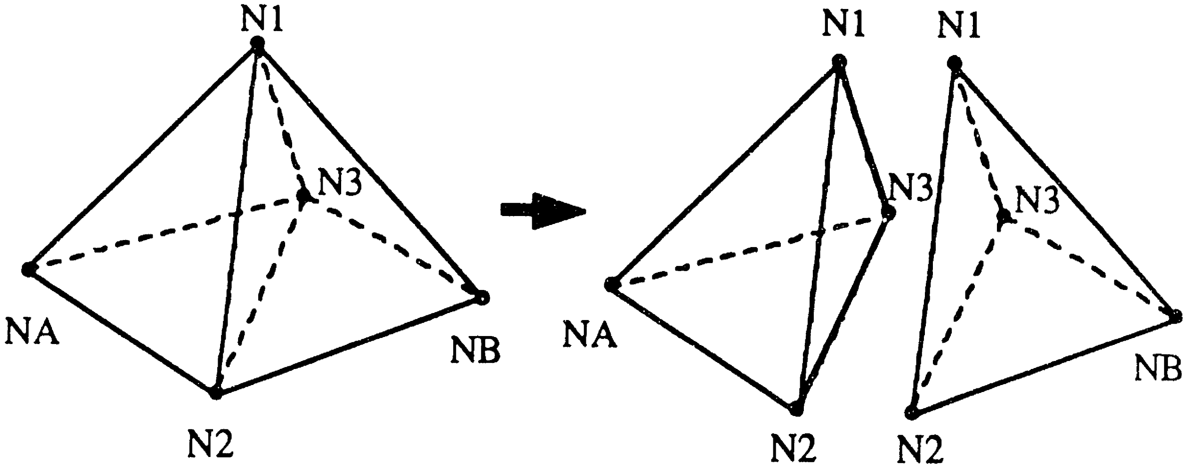
The Type-0 operation is designed to construct the last two or three elements when the number of edges in a loop-boundary reaches 9. With these 9 edges, two kinds of polyhedron (loop-boundary) are possible and the corresponding mesh constructions are shown in Figure 3-20.

As shown in Figure 3-20, the Type-0 operation is a combination of Type-1A and Type-1B operation. Therefore, it also satisfies the derivative of Euler's formula. However, in order to avoid any overlapping between elements a special operation is needed to complete the tetrahedronization. This is acheived by the Type-0 operation.

If the surface triangulation on a component has been performed correctly, the resulting polyhedron with V key nodes, E edges and F faces will satisfy the following Euler's formula,

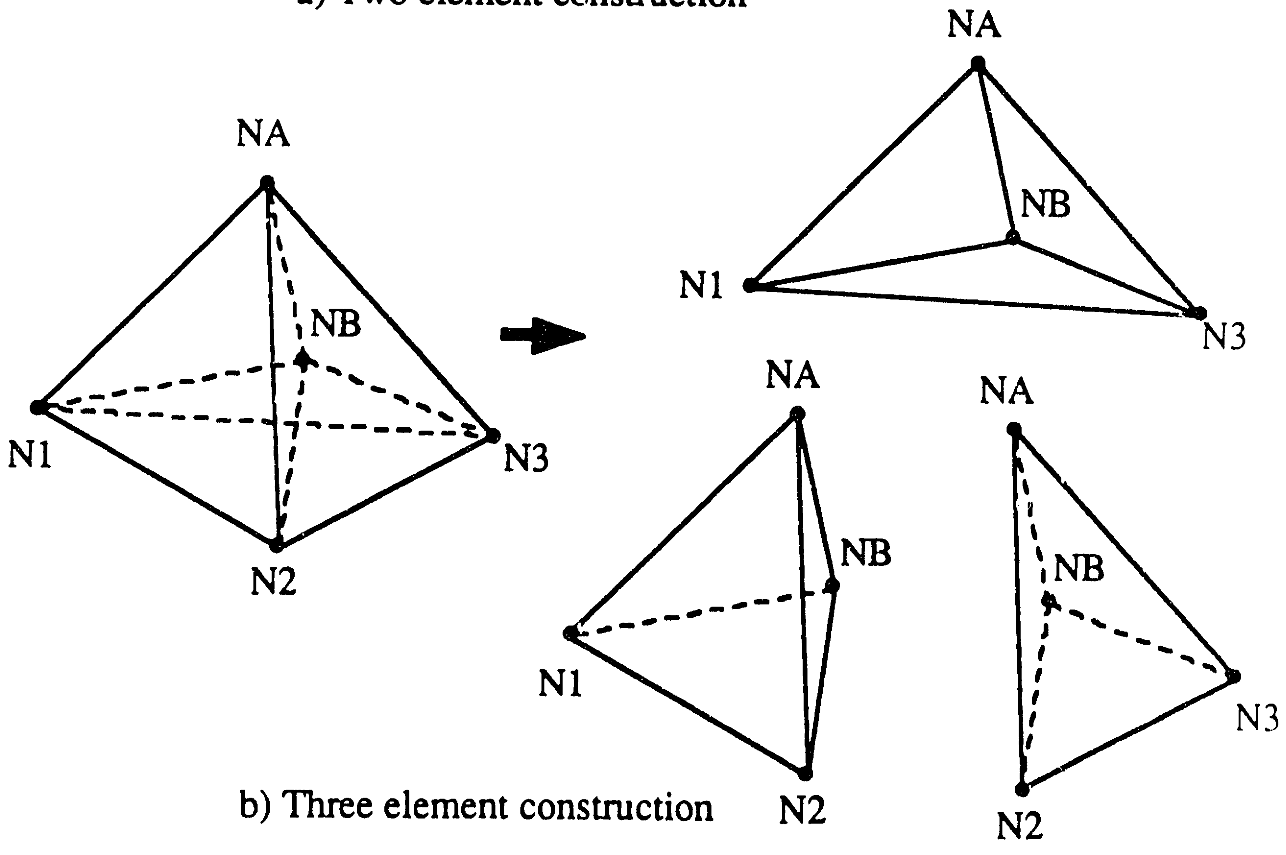
$$V - E + F = 2 \qquad (3\text{--}11)$$

After the surface triangulation process, the resulting polyhedron will have a large number of faces, edges and nodes and the volume triangulation is to reduce the number of faces, edges and nodes by using basic operations and finally



NA,NB ; 3 edges meet at this node

a) Two element construction



b) Three element construction

Figure 3-20: Type-0 operation

to get one tetrahedron (V=4, E=6, F=4). Since all the basic operations satisfy the derivative of Euler's formula, the resulting polyhedron after each operation will also satisfy Euler's formula. Among the basic operations, only the Type-1A operation contributes to reducing the number of faces, edges and nodes (ΔV=-1, ΔE=-3, ΔF=-2), so enough number of Type-1A operations should be performed to complete the volume triangulation process. Type-1B operation(ΔV=0, ΔE=0, ΔF=0) is performed to generate Type-1A edges when no more Type-1A edges are available. The Type-2 operation(ΔV=1, ΔE=3, ΔF=2) is performed to generate a Type-1B operation when either Type-1B or Type-1A edge is not available. Since this operation increases the number of faces, edges and nodes of a loop-boundary, one more Type-1A operation will be necessary to compensate for this adverse effect if one Type-2 operation is performed. Therefore at least three basic operations (Type-1A, Type-1B, Type-2) are required for the volume triangulation process. Starting from an initial polyhedron, the volume triangulation process can always be completed by using these basic operations, providing that there is no overlapping of loop-boundary edges and faces during the volume triangulation process. In order to avoid any overlapping or bottle-neck like region, check processings are designed for our volume triangulation process.

The check processing is necessary during the volume triangulation process for an *overlap check* and *minimum distance check*. In case of the *overlap check*, the new faces of the generated tetrahedral elements are checked whether any overlapping occurs between these faces and the remaining loop-boundary faces. In the actual computer implementation, it is easier to check the overlapping between new faces and the remaining loop-boundary edges rather than to check between new faces and the remaining loop-boundary faces. Hence, the *overlap check* in our discussion is to test whether the remaining loop-boundary edges pierce the new generated faces of each operation. Consider the *overlap check* in Figure 3-21, which is to test whether a loop-boundary edge, a-b, pierces a new face, 1-2-3. The following procedures are used for the *overlap check.*

1. Check whether the coordinate ranges of line a-b oberlap with those of a new face 1-2-3. If any overlapping occurs, go to step-2.
2. Calculate the piercing point between an edge a-b and the plane containing a new face 1-2-3 as follows. The equation of a plane containing a new face 1-2-3 is found to be

$$Z = Ax + By + C \qquad (3\text{-}12)$$

$$\textit{where} \qquad A = \frac{1}{D}(z_{13}y_{23} - y_{13}z_{23})$$

$$B = -\frac{1}{D}(z_{13}x_{23} - x_{13}z_{23})$$

$$C = z_1 - Ax_1 - By_1$$

$$D = x_{13}y_{23} - y_{13}x_{23}$$

$$\textit{and} \qquad x_{13} = x_1 - x_3, \ \textit{etc.}$$

The equation of a line connecting nodes a and b is expressed in parametric form as

$$x = x_a + t \cdot x_{ba}$$

$$y = y_a + t \cdot y_{ba} \qquad (3\text{-}13)$$

$$z = z_a + t \cdot z_{ba}$$

$$\textit{where} \qquad 0 \le t \le 1$$

Thus the piercing point can be found by substuting equation (3-13) into equation (3-12).

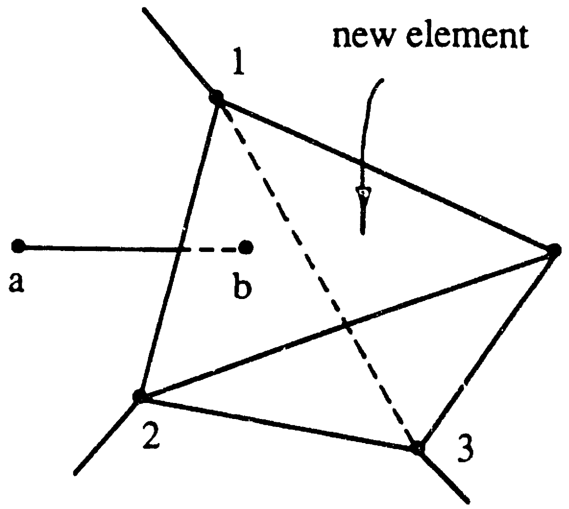


Figure 3-21: Overlap check

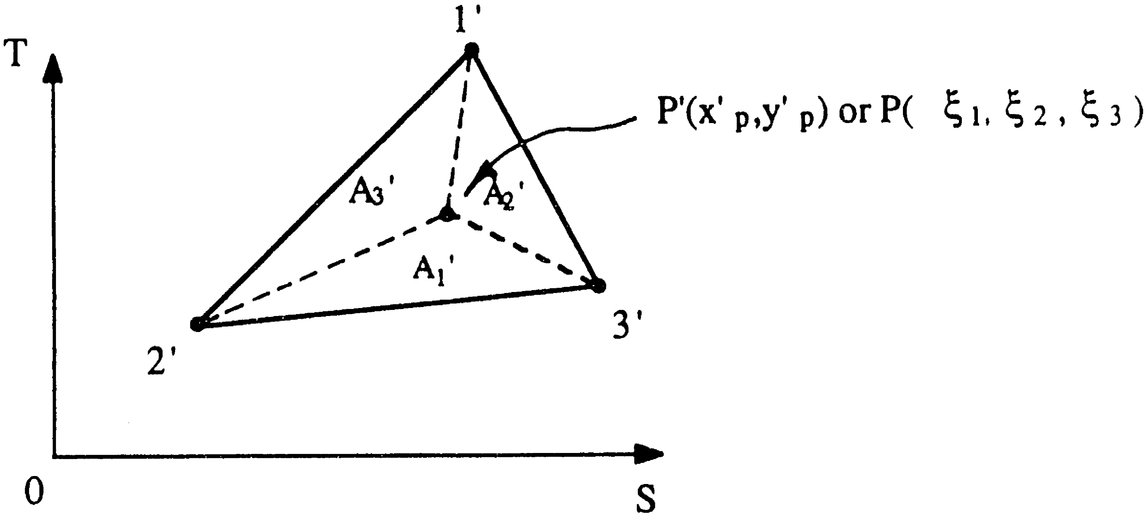


Figure 3-22: Area coordinates of a piercing point

$$t = \frac{z_a - A \cdot x_a - B \cdot y_a - C}{A \cdot x_{ba} + B \cdot y_{ba} - z_{ba}} \qquad (3\text{–}14)$$

If t does not lie in the range zero to one, the piercing point is not valid, so go to step-1 with the next available loop-boundary edge. The piercing point is obtained as

$$\begin{aligned} x_p &= x_a + t \cdot x_{ba} \\ y_p &= y_a + t \cdot y_{ba} \\ z_p &= z_a + t \cdot z_{ba} \end{aligned} \qquad (3\text{–}15)$$

3. Check whether a piercing point, p, is contained within a new face 1-2-3 by using the area coordinates of a triangle[57]. To begin, transform the piercing point, p, and the nodes 1,2 and 3 into a view plane S-T, in which the view direction is parallel to the surface normal vector of a new face 1-2-3 as shown in Figure 3-22. The area coordinates of a piercing point are calculated as follows.

$$\begin{bmatrix} 1 \\ x'_p \\ y'_p \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ x'_1 & x'_2 & x'_3 \\ y'_1 & y'_2 & y'_3 \end{bmatrix} \begin{bmatrix} \zeta_1 \\ \zeta_2 \\ \zeta_3 \end{bmatrix} \qquad (3\text{-}16)$$

or

$$\begin{bmatrix} \zeta_1 \\ \zeta_2 \\ \zeta_3 \end{bmatrix} = \frac{1}{D'} \begin{bmatrix} x'_2 y'_3 - x'_3 y'_2 & y_{23}' & x_{32}' \\ x'_3 y'_1 - x'_1 y'_3 & y_{31}' & x_{13}' \\ x'_1 y'_2 - x'_2 y'_1 & y_{12}' & x_{21}' \end{bmatrix} \begin{bmatrix} 1 \\ x'_p \\ y'_p \end{bmatrix} \qquad (3\text{-}17)$$

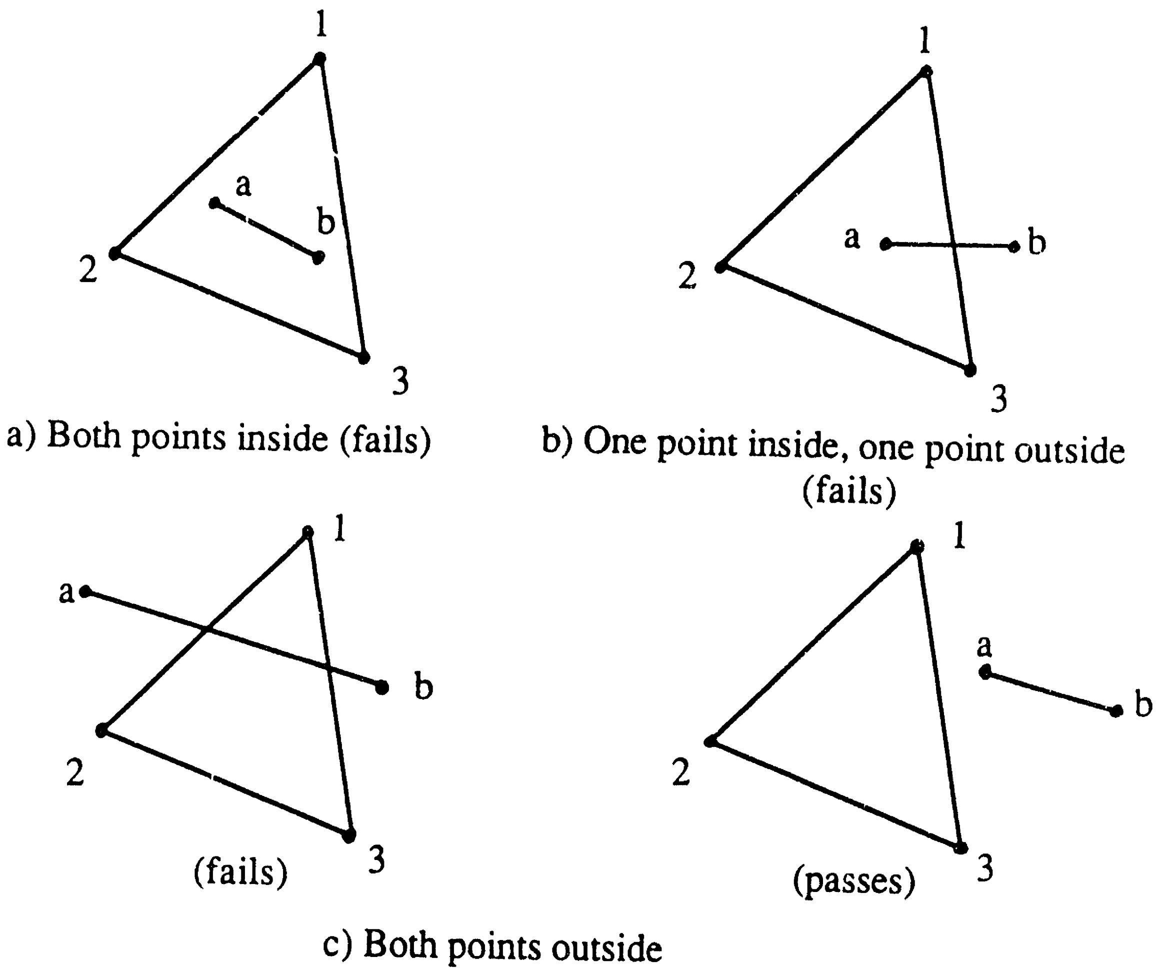
where $D' = x_{13}' y_{23}' - y_{13}' x_{23}'$

If $\min[\zeta_1, \zeta_2, \zeta_3]$ and $\max[\zeta_1, \zeta_2, \zeta_3]$ both lie in the range zero to one, the piercing point is contained within the triangle, i.e. a loop-boundary edge pierces a new generated face. Therefore, the overlap check fails in this operation.

4. If an edge lies in the same plane containing a new face, area coordinates are used for *in* and *out* test of nodes a and b w.r.t. a new face and a loop-boundary edge a-b is also tested for the intersection with the edges of a new face as shown in Figure 3-23.

For the overlap check, one new face is tested in Type-1A operation and two new faces are tested in Type-1B operation and four new faces are tested in Type-2 operation.

In the case of the *minimum distance check*, a new key node is tested whether it has enough distance to the remaining loop-boundary faces in order to construct well-conditioned elements. The distance from a new key node to a loop-boundary face can be checked by transforming a new key node into a view plane of a loop-boundary face, in which the view direction is parallel to the surface normal of the face and comparing the relative depths in the view direction as shown in Figure 3-24. The minimum distance requirement to a loop-boundary face changes according to a parameter, *ifail*, used in *adaptive* Type-2 operation. As shown in Figure 3-10, the height of an equilateral tetrahedron is represented as a function of the face area ($h = 1.24\sqrt{A}$). Initially, the minimum distance to a loop-boundary face is set approximately to a half of the height, h, of an assumed desired tetrahedron and a parameter, *ifail*, is introduced to adjust this distance as follows.



a) Both points inside (fails)

b) One point inside, one point outside (fails)

(fails)

(passes)

c) Both points outside

Figure 3-23: Intersection test in overlap check

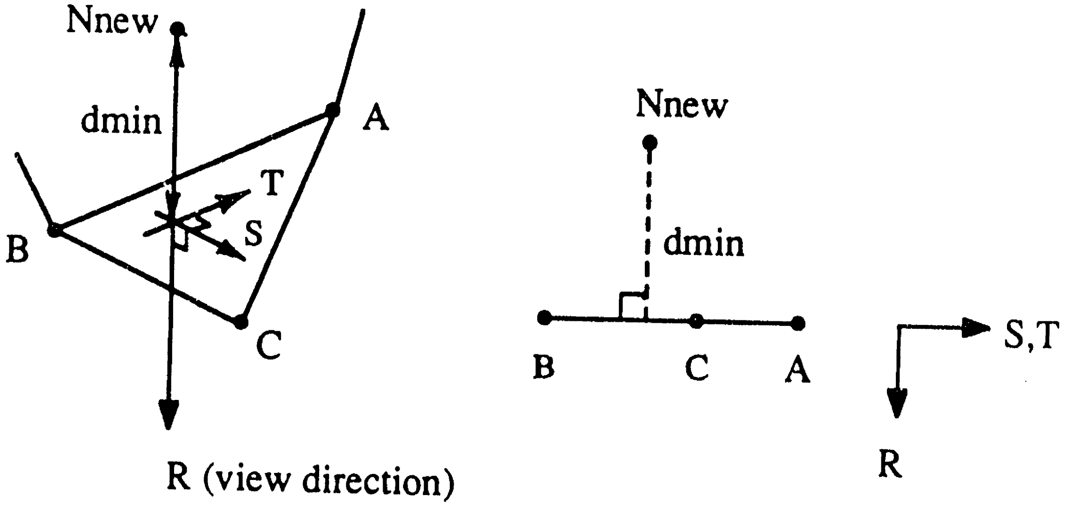


Figure 3-24: Minimum distance check

$$d_{min} = \frac{0.6}{1 + ifail}\sqrt{A} \qquad (3\text{-}18)$$

$$ifail = 0 \; ; \; d_{min} = 0.6\sqrt{A} \quad (\approx 0.48h)$$

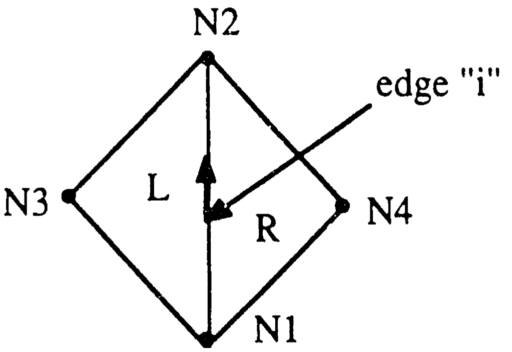
$$ifail = 1 \; ; \; d_{min} = 0.3\sqrt{A} \quad (\approx 0.24h)$$

$$ifail = 2 \; ; \; d_{min} = 0.2\sqrt{A} \quad (\approx 0.16h)$$

and etc.

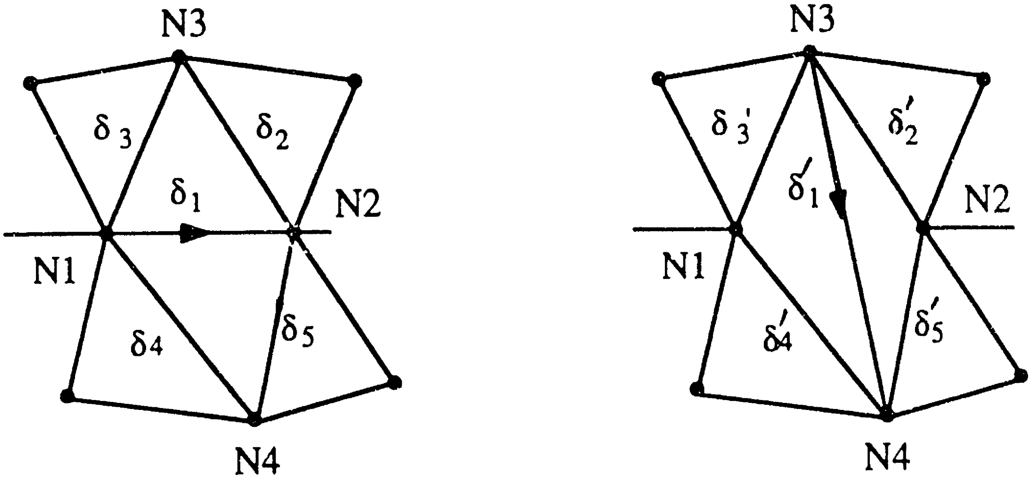
Similar to the case of surface triangulation process, the key issues in volume triangulation process are i) how to define Type-1A, Type-1B and Type-2 corner edges among the loop-boundary edges and ii) in which order the basic operations should be performed. The following heuristic rules are used in our discussion.

1. A Type-1A edge is an edge which has one common surrounding face as shown in Figure 3-12, which is equivalent to the condition that three edges meet at one node, and the edge angle is less than or equal to 120°.
2. A Type-1B corner edge is a loop-boundary edge where there is no common surrounding face, i.e. more than four edges meet at one node, and the edge angle is less than 85° ($\delta_i \geq 1.8$) or 100° ($\delta_i < 1.8$) depending on the face area ratio , δ_i and the surrounding face area ratios do not change drastically due to the Type-1B operation as shown in Figure 3-25. In addition, the new generated edge due to Type-1B operation should not be an existing edge in order to avoid splitting a loop-boundary domain into two subdomains as shown in Figure 3-26. And the new generated edge should not be in any face of a loop-boundary in order to avoid overlapping between a new edge and the exisiting edges as shown in Figure 3-27. In Figure 3-27, if a Type-1B operation is performed on an edge, KE, the new generated edge N3-N4 overlaps with the existing edge N1-NS. This is a typical case when no more Type-1A or Type-1B operation is possible on all 9 edges, so Type-2 operation which generates a new key node inside a loop-boundary is necessary.



$$\delta_i = \max\left(\frac{A_L}{A_R}, \frac{A_R}{A_L}\right) : \text{area ratio}$$

A_L, A_R : areas of adjacent faces



before operation;

$\delta\ old = \max(\delta_1, \delta_2, \delta_3, \delta_4, \delta_5)$

after operation;

$\delta\ new = \max(\delta_1', \delta_2'\ \delta_3', \delta_4'\ \delta_5')$

```
If $\phi_i \le 60°$ then type-1B edge
Elseif $60° < \phi_i \le 85° \sim 100°$ then
        if $\delta_{old} \le 1.2$ then
                if $\delta_{new} \le 1.7\, \delta_{old}$ then type-1B edge
        elseif $1.2 < \delta_{old} \le 1.5$ then
                if $\delta_{new} \le 1.3\, \delta_{old}$ then type-1B edge
        else
                if $\delta_{new} \le 1.2\, \delta_{old}$ then type-1B edge
        endif
Else
Endif
```

Figure 3-25: Type-1B edge decision

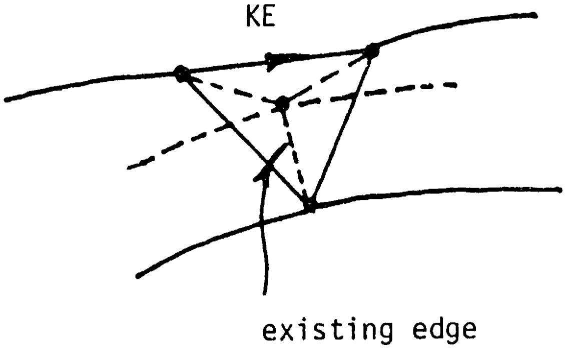


Figure 3-26: Example of an existing edge

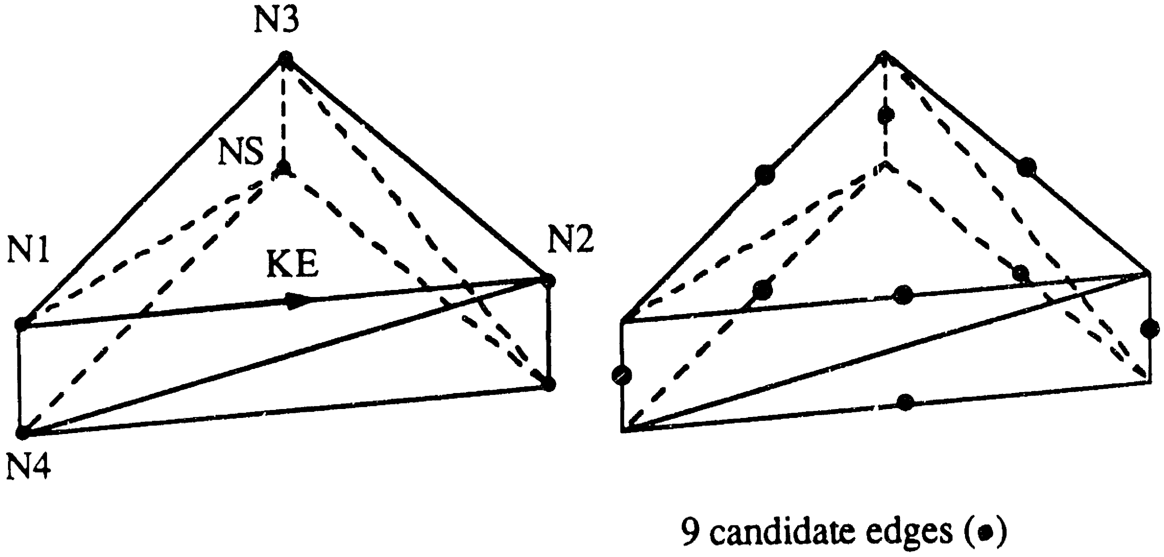


Figure 3-27: Example of overlapping edges in Type-1B operation

3. A Type-2 corner edge is an edge which is neither Type-1A edge nor Type-1B edge and the edge angle is less than 175°, suitable for generating two well-conditioned elements.

4. Among different types of operation, the order of operation is determined as follows.

   - Type-1A operation
   - Type-1B operation
   - Type-2 operation

5. Among the same types of operations, the order of operation is also an important factor especially in the case of Type-2 operation.

6. Type-1A and Type-1B edges are sorted to decide the order of operations considering the following factors successively.

   - Low level
   - Small edge angle, ϕ_i
   - Small adjacent face area, A_i
   - Large adjacent face area ratio, δ_i

7. Type-2 edges are sorted to decide the order of operations considering the following factors successively.

   - Low level
   - Large adjacent face area ratio, δ_i
   - Small adjacent face area, A_i
   - Small edge angle, ϕ_i

After the volume triangulation process has been completed for a given component, a subregion domain, the resulting mesh is finally improved by the application of the smoothing technique described in Chapter 2. It also has been observed that usually 5 to 10 iterations are enough to converge to a given tolerance of 1/500 L_{system} (characteristic length of the system).

All the above details are summarized in the flow chart of the entire volume triangulation procedures shown in Figure 3-28. An example of the volume triangulation scheme developed in this research is shown in Figure 3-29 to

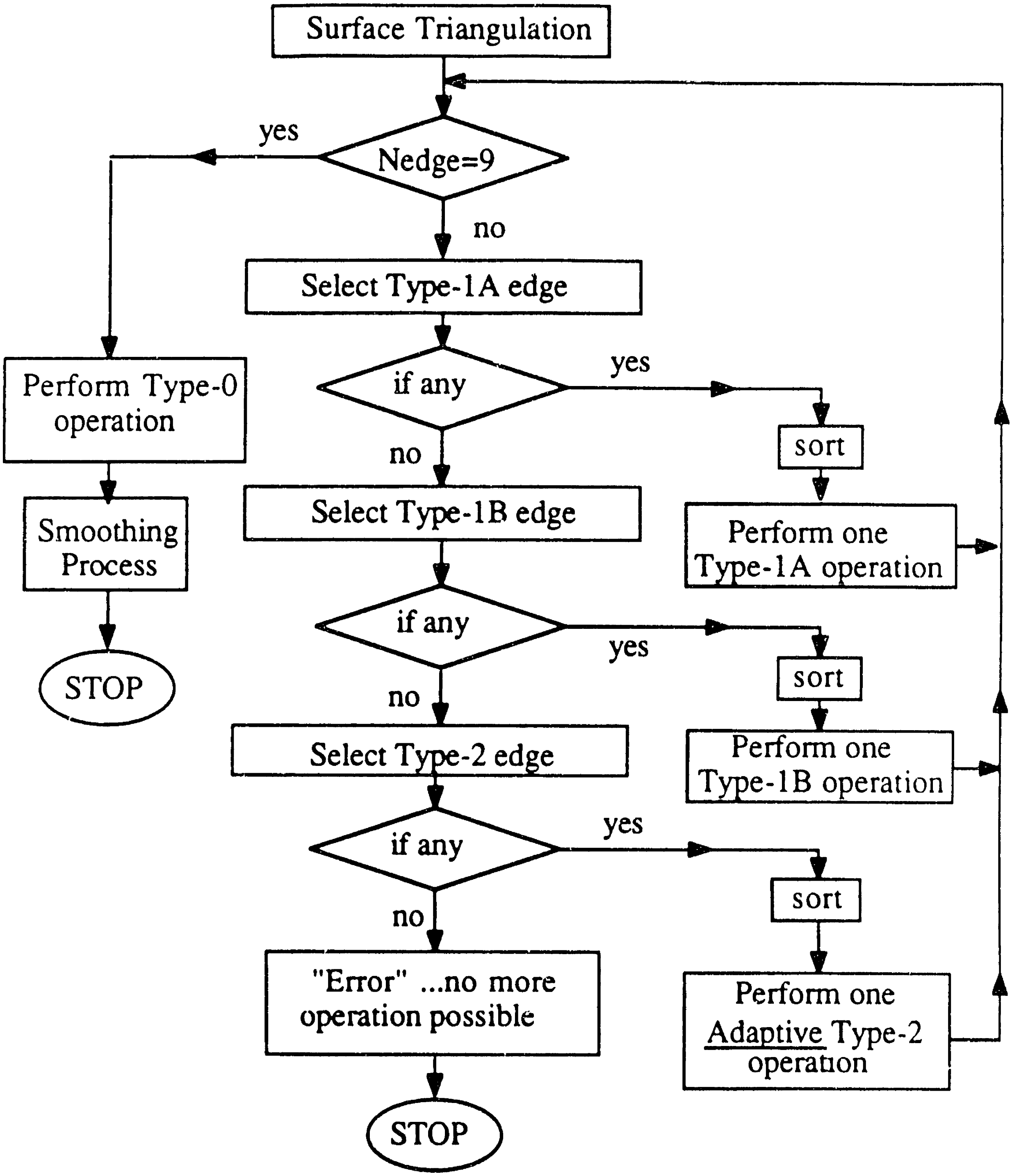


Figure 3-28: Flow chart of the volume triangulation process

Figure 3-33. In Figure 3-29 a) we consider a cylinder with the loop-faces and loop-edges defined for the mesh generation. The example of the input data is shown in Figure 3-29 b) and the mesh is obtained as in Figure 3-29 c).

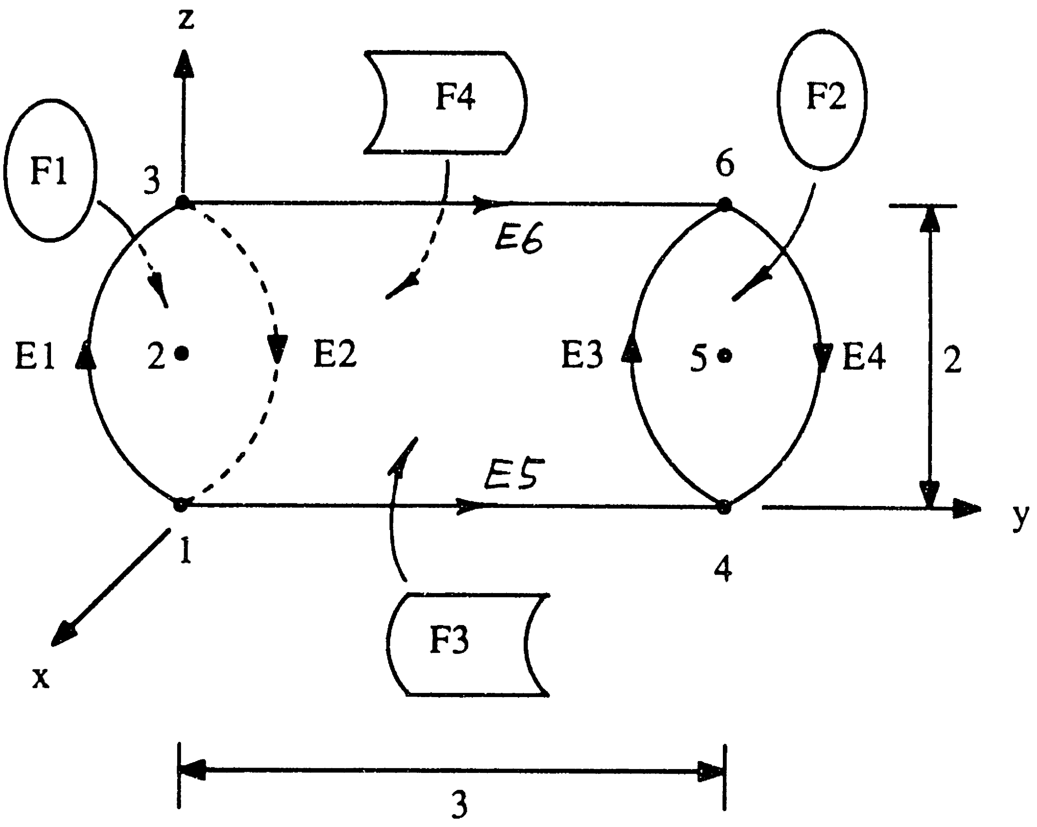
Since only straight-sided tetrahedral elements are employed for this mesh generation, the curved surface of the cylinder is not accurately modeled, which results in 6.5 % error in the volume. We also see that after the smoothing process, the minimum dihedral angle in the mesh has increased from 18.2° to 31.0°. The total number of elements constructed is 279, which compares well with the estimated number, $N_{estimate}$ = 314, obtained by assuming a relatively small size of element as a reference.

Another example is considered for the mesh generation, namely the example of a cube as shown in Figure 3-30. The number of elements in the cube is 92, which is within an estimated range ($N_{estimate}$ = 72 - 162), and the minimum dihedral angle in the mesh is obtained as 42.9° after smoothing. Hence, we see that our mesh generation algorithm works very well in this example.

Graded meshes are also constructed in Figure 3-31 and Figure 3-32 for the same cube, which show reasonable results. The minimum dihedral angle in the mesh is obtained as small as 9.7° due to the drastic change in the element size assigned as an input.

However, if we consider a fine mesh construction as shown in Figure 3-33, the result is not quite as satisfactory as that of the coarse mesh in Figure 3-30. The number of elements constructed is 829, more than the estimated value, $N_{estimate}$ = 750, obtained by assuming a small size element. The minimum dihedral angle after smoothing is 19.8°.

Therefore, the suggested algorithm works very well for a relatively

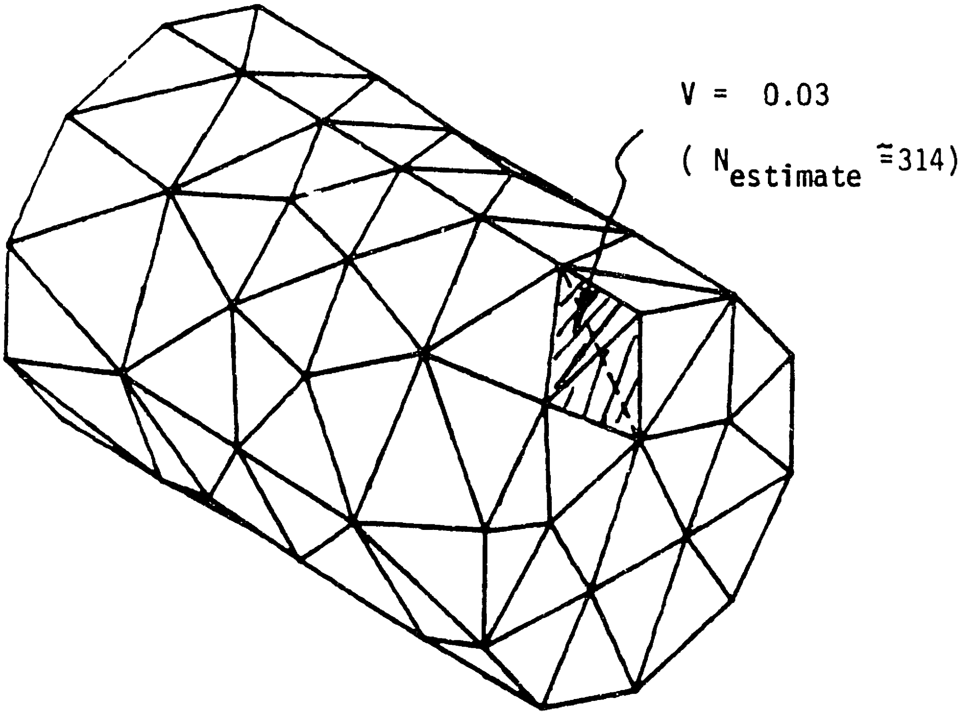


Volume = 9.42

Figure 3-29 a): Cylinder with loop-faces and loop-edges

```
 1     4     6     4     6   3.0
 1
 4     6
 1     2     3     4                          object
 1     2     3     4     5     6
 1     1     0     2
 1     2
 2     1     0     2                          loop-face
-3    -4
 3     2     1     4     1
-1     5     3    -6
 4     2     1     4     2
 4    -5    -2     6
 1     1     3     1     5   1.0
 2     3     1     2     5   1.0
 3     4     6     3     5   1.0              loop-edge
 4     6     4     4     5   1.0
 5     1     4     0     6   1.0
 6     3     6     0     6   1.0
 1     2     1
 2     2     1                                circle
 3     5    -1
 4     5    -1
 1           0           0         0
 2           0           0         1
 3           0           0         2
 4           0           3         0          node
 5           0           3         1
 6           0           3         2
 1           0          -1         0
 2           0           1         0          cylinder
```

Figure 3-29 b): Input data for mesh generation of a cylinder



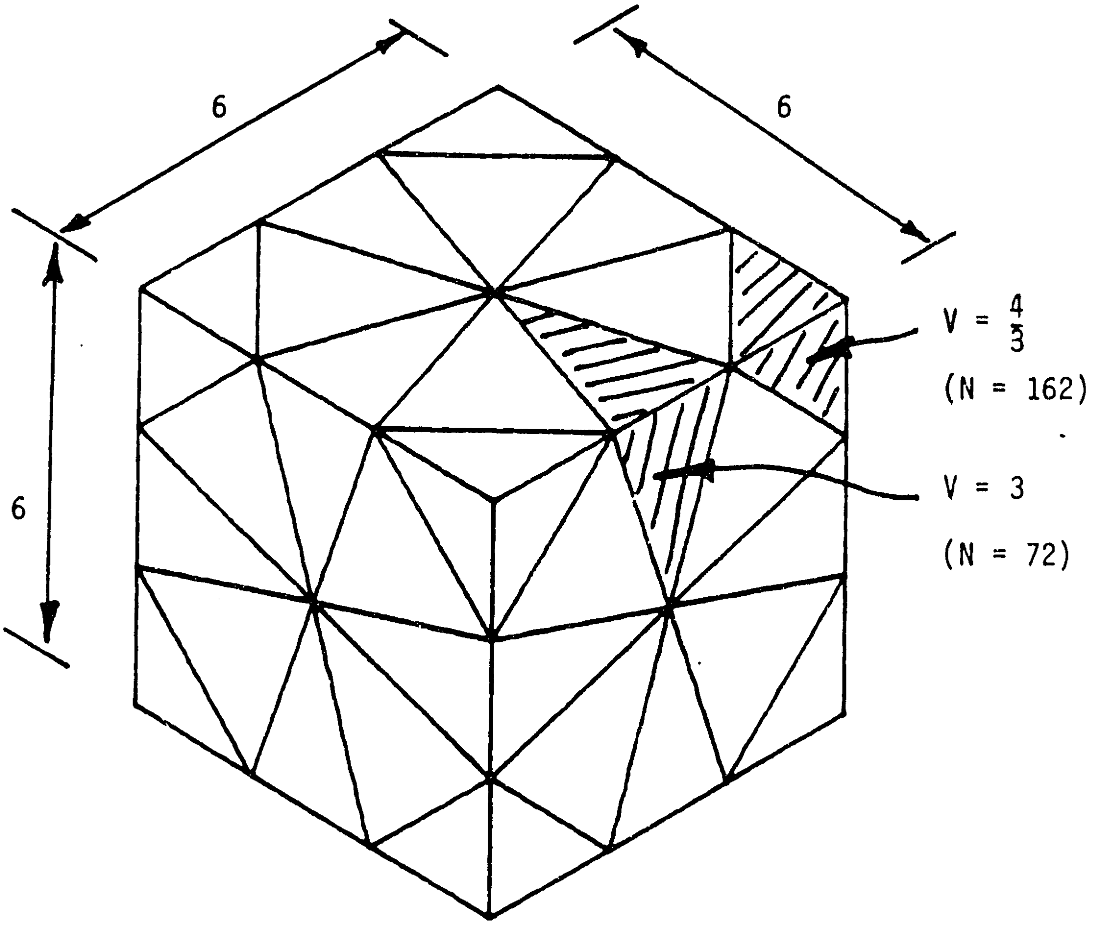
```
 *** BEFORE SMOOTHING ; IOBJ =     1
   MIN VOL  =    0.010   EL NO =  225   CHARACT. LENG =    0.436
   MAX VOL  =    0.069   EL NO =   84   CHARACT. LENG =    0.837
   AVG VOL  =    0.032     TOT VOL = 0.881E+01
   MIN ANGLE = 18.2      EL NO =  212

** SMOOTHING HAS CONVERGED : NO. OF ITERATION =    6

 *** AFTER SMOOTHING ; IOBJ =     1
   MIN VOL  =    0.013   EL NO =  273   CHARACT. LENG =    0.477
   MAX VOL  =    0.065   EL NO =  183   CHARACT. LENG =    0.822
   AVG VOL  =    0.032     TOT VOL = 0.881E+01
   MIN ANGLE = 31.0      EL NO =   35

         NUMEL =  279
         NUMNP =  534
         NUMFACE =  626
         NUMEDG =  439
```

Figure 3-29 c): 279 element mesh for a cylinder



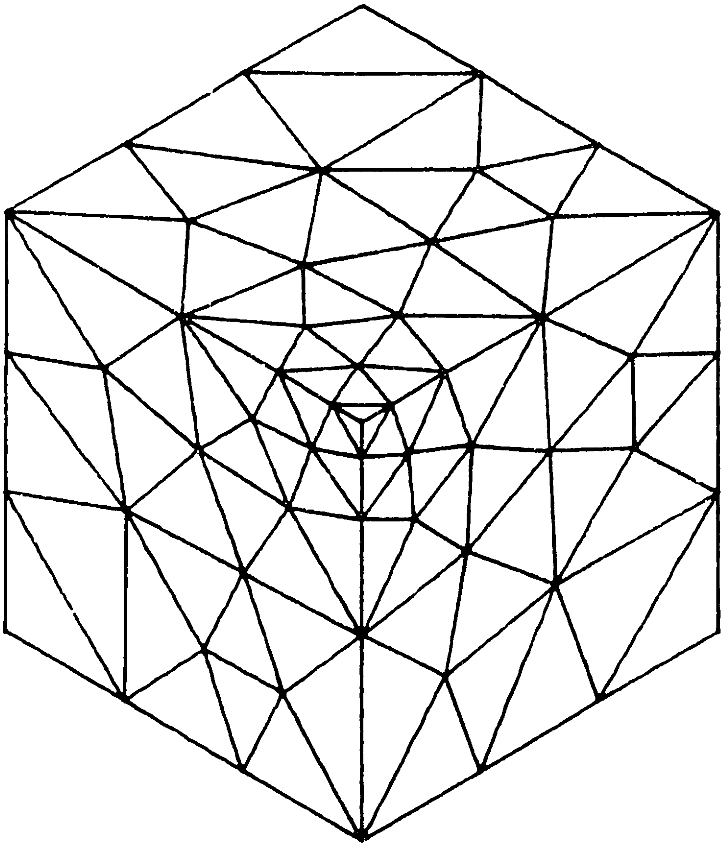
```
 *** BEFORE SMOOTHING ; IOBJ =     1
   MIN VOL =    1.326   EL NO =   69   CHARACT. LENG =    2.241
   MAX VOL =    5.758   EL NO =   90   CHARACT. LENG =    3.656
   AVG VOL =    2.348      TOT VOL = 0.216E+03
   MIN ANGLE = 28.2      EL NO =   44

** SMOOTHING HAS CONVERGED : NO. OF ITERATION =     6

 *** AFTER SMOOTHING ; IOBJ =     1
   MIN VOL =    1.333   EL NO =    1   CHARACT. LENG =    2.245
   MAX VOL =    4.520   EL NO =   90   CHARACT. LENG =    3.373
   AVG VOL =    2.348      TOT VOL = 0.216E+03
   MIN ANGLE = 42.9      EL NO =   36

         NUMEL =    92
         NUMNP =   221
         NUMFACE =   220
         NUMEDG =   174
```

Figure 3-30: Uniform coarse mesh for a cube



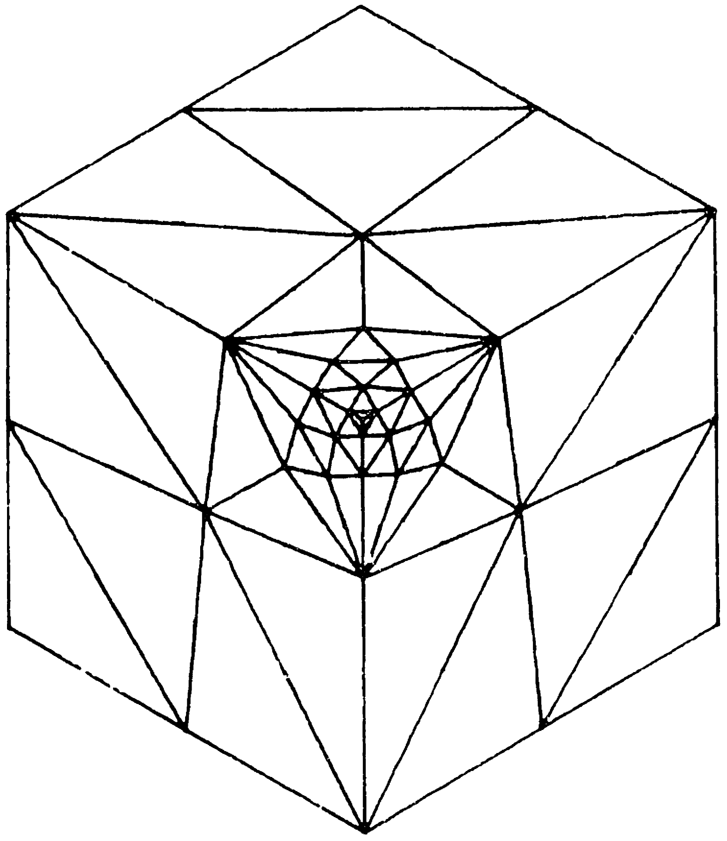
```
 *** BEFORE SMOOTHING ; IOBJ =     1
   MIN VOL =    0.022   EL NO =    5    CHARACT. LENG =   0.567
   MAX VOL =    5.993   EL NO =  284    CHARACT. LENG =   3.706
   AVG VOL =    0.753     TOT VOL = 0.216E+03
   MIN ANGLE = 21.6      EL NO =  258

** SMOOTHING HAS CONVERGED : NO. OF ITERATION =   11

 *** AFTER SMOOTHING ; IOBJ =     1
   MIN VOL =    0.022   EL NO =    5    CHARACT. LENG =   0.567
   MAX VOL =    5.993   EL NO =  284    CHARACT. LENG =   3.706
   AVG VOL =    0.753     TOT VOL = 0.216E+03
   MIN ANGLE = 31.1      EL NO =   63

         NUMEL =  287
         NUMNP =  537
         NUMFACE =  637
         NUMEDG =  443
```

Figure 3-31: Graded mesh-1 for a cube



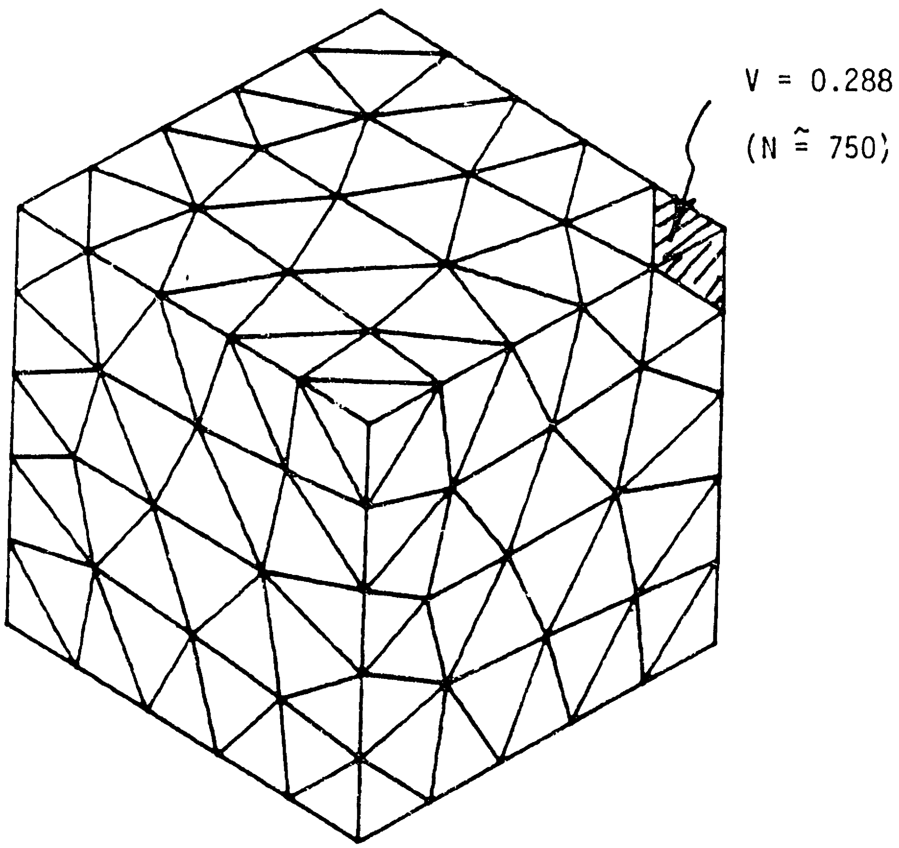
```
*** BEFORE SMOOTHING ; IOBJ =      1
   MIN VOL =    0.002   EL NO =    5   CHARACT. LENG =   0.265
   MAX VOL =   18.000   EL NO =   32   CHARACT. LENG =   5.346
   AVG VOL =    1.149     TOT VOL = 0.216E+03
   MIN ANGLE =   7.1     EL NO =  125

** SMOOTHING HAS CONVERGED : NO. ITERATION =    10

 *** AFTER SMOOTHING ; IOBJ =      1
   MIN VOL =    0.002   EL NO =    5   CHARACT. LENG =   0.265
   MAX VOL =   18.000   EL NO =   32   CHARACT. LENG =   5.346
   AVG VOL =    1.149     TOT VOL = 0.216E+03
   MIN ANGLE =   9.7     EL NO =  125

          NUMEL  =   188
          NUMNP  =   354
          NUMFACE =   415
          NUMEDG =   290
```

Figure 3-32: Graded mesh-2 for a cube



```
 *** BEFORE SMOOTHING ; IOBJ =     1
   MIN VOL =    0.021    EL NO =  678   CHARACT. LENG =    0.565
   MAX VOL =    0.755    EL NO =  450   CHARACT. LENG =    1.857
   AVG VOL =    0.261      TOT VOL = 0.216E+03
   MIN ANGLE = 12.3      EL NO =  741

** SMOOTHING HAS CONVERGED : NO. OF ITERATION =    7

 *** AFTER SMOOTHING ; IOBJ =     1
   MIN VOL =    0.028    EL NO =  646   CHARACT. LENG =    0.618
   MAX VOL =    0.733    EL NO =  367   CHARACT. LENG =    1.839
   AVG VOL =    0.261      TOT VOL = 0.216E+03
   MIN ANGLE = 19.8      EL NO =  510

         NUMEL =  829
         NUMNP = 1372
         NUMFACE = 1766
         NUMEDG = 1154
```

Figure 3-33: Fine uniform mesh for a cube

coarse mesh generation, but for a fine mesh generation the result is not as satisfactory.

3.5 Error Analysis

An error analysis for three-dimensional problems, i.e. a computation of error indicators, has been implemented in our research, although an adaptive refinement process is not included. The error indicator used in two-dimensional problems is written for three-dimensional problems as follows,

$$\eta_m = \frac{h_m}{E} \int_{V_m} (R_x{}^2 + R_y{}^2 + R_z{}^2)\, dV \qquad (3.19)$$

As a reference value, the total strain energy of the system has been used with the following modification,

$$U_{reference} = \frac{U_{total}}{L_{system}} \qquad (3\text{–}20)$$

where $U_{reference}$; *reference strain energy per unit length*
U_{total} ; *total strain energy of the system*
L_{system} ; *characteristic length of the system*

As a criterion for further refinement , a non-dimensionalized error indicator of each element defined as follows could be used,

$$\varepsilon_m = \frac{\eta_m}{U_{reference}} \qquad (3\text{–}21)$$

The total error indicator of the system is found to be

$$\varepsilon_{total} = \sum_{m=1}^{N} \varepsilon_m \tag{3-22}$$

The equilibrium equations for three-dimensional problems are

$$\begin{aligned} R_x &= \frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z} + f_x = 0 \\ R_y &= \frac{\partial \tau_{yx}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \tau_{yz}}{\partial z} + f_y = 0 \\ R_z &= \frac{\partial \tau_{zx}}{\partial x} + \frac{\partial \tau_{zy}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z} + f_z = 0 \end{aligned} \tag{3-23}$$

The stress-strain relationship in three-dimensional problem is written [58] as

$$\begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \tau_{xy} \\ \tau_{yz} \\ \tau_{zx} \end{bmatrix} = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1 & \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & & & \\ \frac{\nu}{1-\nu} & 1 & \frac{\nu}{1-\nu} & & & \\ \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 1 & & & \\ & & & \frac{1-2\nu}{2(1-\nu)} & & \\ & & & & \frac{1-2\nu}{2(1-\nu)} & \\ & & & & & \frac{1-2\nu}{2(1-\nu)} \end{bmatrix} \begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{bmatrix}$$

(3–24)

Using the equations (3-23) and (3-24) and the strain-displacement relationships, the residuals are obtained as summarized in Appendix B.

By the chain rule, the relationship in Table 3-1 is obtained and is used to calculate the second derivative operators $\partial^2/\partial x^2$, $\partial^2/\partial y^2$, $\partial^2/\partial z^2$, $\partial^2/\partial x \partial y$, $\partial^2/\partial y \partial z$ and $\partial^2/\partial z \partial x$. When computing the error indicators by Gauss integration, the residuals should be calculated at every integration point of each element. Hence, the equations in Table 3-1 must be used at every integration point of each element. In order to reduce the amount of computations involved, only straight-sided tetrahedra with mid-side nodes located at the middle of their edges are used in the error analysis. In this case, the Jacobian matrix $\underline{J}$ is constant and the computation in Table 3-1 is considerably simplified.

Consider a 10-node tetrahedral element and the corresponding interpolation functions as shown in Figure 3-34.[58]

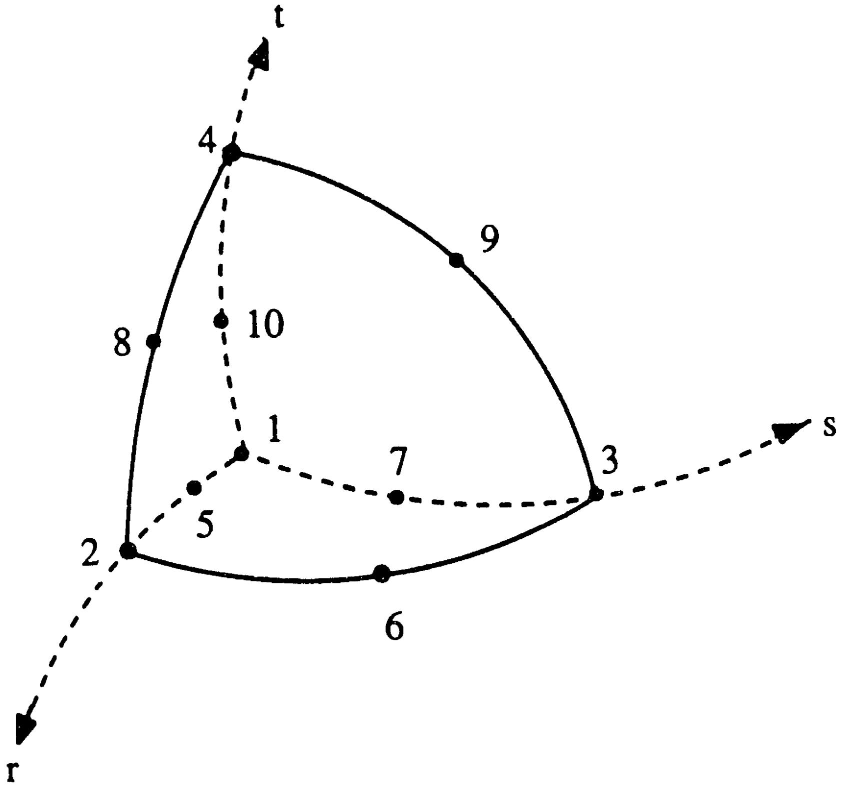
If we consider a straight-sided tetrahedral element with mid-side nodes located at the middle of their edges, the derivatives of the interpolation functions are obtained as shown in Table 3-2, and the $\underline{J}$ matrix is constant through the element, as follows.

$$\underline{J} = \begin{bmatrix} \begin{bmatrix} J_{11} \end{bmatrix}_{3x3} & 0 \\ \begin{bmatrix} J_{21} \end{bmatrix}_{6x3} & \begin{bmatrix} J_{22} \end{bmatrix}_{6x6} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} J_{11} \end{bmatrix}_{3x3} & 0 \\ 0 & \begin{bmatrix} J_{22} \end{bmatrix}_{6x6} \end{bmatrix} \qquad (3\text{-}25)$$

where

$$
\begin{bmatrix}
\frac{\partial}{\partial r} \\
\frac{\partial}{\partial s} \\
\frac{\partial}{\partial t} \\
\frac{\partial^2}{\partial r^2} \\
\frac{\partial^2}{\partial s^2} \\
\frac{\partial^2}{\partial t^2} \\
\frac{\partial^2}{\partial r \partial s} \\
\frac{\partial^2}{\partial s \partial t} \\
\frac{\partial^2}{\partial r \partial t}
\end{bmatrix}
=
\begin{bmatrix}
\frac{\partial x}{\partial r} & \frac{\partial y}{\partial r} & \frac{\partial z}{\partial r} & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{\partial x}{\partial s} & \frac{\partial y}{\partial s} & \frac{\partial z}{\partial s} & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{\partial x}{\partial t} & \frac{\partial y}{\partial t} & \frac{\partial z}{\partial t} & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{\partial^2 x}{\partial r^2} & \frac{\partial^2 y}{\partial r^2} & \frac{\partial^2 z}{\partial r^2} & (\frac{\partial x}{\partial r})^2 & (\frac{\partial y}{\partial r})^2 & (\frac{\partial z}{\partial r})^2 & 2\frac{\partial x}{\partial r}\frac{\partial y}{\partial r} & 2\frac{\partial y}{\partial r}\frac{\partial z}{\partial r} & 2\frac{\partial z}{\partial r}\frac{\partial x}{\partial r} \\
\frac{\partial^2 x}{\partial s^2} & \frac{\partial^2 y}{\partial s^2} & \frac{\partial^2 z}{\partial s^2} & (\frac{\partial x}{\partial s})^2 & (\frac{\partial y}{\partial s})^2 & (\frac{\partial z}{\partial s})^2 & 2\frac{\partial x}{\partial s}\frac{\partial y}{\partial s} & 2\frac{\partial y}{\partial s}\frac{\partial z}{\partial s} & 2\frac{\partial z}{\partial s}\frac{\partial x}{\partial s} \\
\frac{\partial^2 x}{\partial t^2} & \frac{\partial^2 y}{\partial t^2} & \frac{\partial^2 z}{\partial t^2} & (\frac{\partial x}{\partial t})^2 & (\frac{\partial y}{\partial s})^2 & (\frac{\partial z}{\partial t})^2 & 2\frac{\partial x}{\partial t}\frac{\partial y}{\partial t} & 2\frac{\partial y}{\partial t}\frac{\partial z}{\partial t} & 2\frac{\partial z}{\partial t}\frac{\partial x}{\partial t} \\
\frac{\partial^2 x}{\partial r \partial s} & \frac{\partial^2 y}{\partial r \partial s} & \frac{\partial^2 z}{\partial r \partial s} & \frac{\partial x}{\partial r}\frac{\partial x}{\partial s} & \frac{\partial y}{\partial r}\frac{\partial y}{\partial s} & \frac{\partial z}{\partial r}\frac{\partial z}{\partial s} & \frac{\partial x}{\partial r}\frac{\partial y}{\partial s} + \frac{\partial x}{\partial s}\frac{\partial y}{\partial r} & \frac{\partial y}{\partial r}\frac{\partial z}{\partial s} + \frac{\partial y}{\partial s}\frac{\partial z}{\partial r} & \frac{\partial x}{\partial r}\frac{\partial z}{\partial s} + \frac{\partial x}{\partial s}\frac{\partial z}{\partial r} \\
\frac{\partial^2 x}{\partial s \partial t} & \frac{\partial^2 y}{\partial s \partial t} & \frac{\partial^2 z}{\partial s \partial t} & \frac{\partial x}{\partial s}\frac{\partial x}{\partial t} & \frac{\partial y}{\partial s}\frac{\partial y}{\partial t} & \frac{\partial z}{\partial s}\frac{\partial z}{\partial t} & \frac{\partial x}{\partial s}\frac{\partial y}{\partial t} + \frac{\partial x}{\partial t}\frac{\partial y}{\partial s} & \frac{\partial y}{\partial s}\frac{\partial z}{\partial t} + \frac{\partial y}{\partial t}\frac{\partial z}{\partial s} & \frac{\partial x}{\partial s}\frac{\partial z}{\partial t} + \frac{\partial x}{\partial t}\frac{\partial z}{\partial s} \\
\frac{\partial^2 x}{\partial r \partial t} & \frac{\partial^2 y}{\partial r \partial t} & \frac{\partial^2 z}{\partial r \partial t} & \frac{\partial x}{\partial r}\frac{\partial x}{\partial t} & \frac{\partial y}{\partial r}\frac{\partial y}{\partial t} & \frac{\partial z}{\partial r}\frac{\partial z}{\partial t} & \frac{\partial x}{\partial r}\frac{\partial y}{\partial t} + \frac{\partial x}{\partial t}\frac{\partial y}{\partial r} & \frac{\partial y}{\partial r}\frac{\partial z}{\partial t} + \frac{\partial y}{\partial t}\frac{\partial z}{\partial r} & \frac{\partial x}{\partial r}\frac{\partial z}{\partial t} + \frac{\partial x}{\partial t}\frac{\partial z}{\partial r}
\end{bmatrix}
\begin{bmatrix}
\frac{\partial}{\partial x} \\
\frac{\partial}{\partial y} \\
\frac{\partial}{\partial z} \\
\frac{\partial^2}{\partial x^2} \\
\frac{\partial^2}{\partial y^2} \\
\frac{\partial^2}{\partial z^2} \\
\frac{\partial^2}{\partial x \partial y} \\
\frac{\partial^2}{\partial y \partial z} \\
\frac{\partial^2}{\partial z \partial x}
\end{bmatrix}
$$

Table 3-1: Relationships between the derivatives



$$h_1 = (1-r-s-t)(1-2r-2s-2t)$$
$$h_2 = r(-1+2r)$$
$$h_3 = s(-1+2s)$$
$$h_4 = t(-1+2t)$$
$$h_5 = 4r(1-r-s-t)$$
$$h_6 = 4rs$$
$$h_7 = 4s(1-r-s-t)$$
$$h_8 = 4rt$$
$$h_9 = 4st$$
$$h_{10} = 4t(1-r-s-t)$$

Figure 3-34: Interpolation functions for a 10-node tetrahedral element [58]

i	$h_{i,r}$	$h_{i,s}$	$h_{i,t}$	$h_{i,rr}$	$h_{i,ss}$	$h_{i,tt}$	$h_{i,rs}$	$h_{i,st}$	$h_{i,rt}$
1	-3+(r+s+t)	-3+(r+s+t)	-3+(r+s+t)	4	4	4	4	4	4
2	-1+4r	0	0	4	0	0	0	0	0
3	0	-1+4r	0	0	4	0	0	0	0
4	0	0	-1+4r	0	0	4	0	0	0
5	4(1-2r-s-t)	-4r	-4r	-8	0	0	-4	0	-4
6	4s	4r	0	0	0	0	4	0	0
7	-4s	4(1-r-2s-t)	-4s	0	-8	0	-4	-4	0
8	4t	0	4r	0	0	0	0	0	4
9	0	4t	4s	0	0	0	0	4	0
10	-4t	-4t	4(1-r-s-2t)	0	0	-8	0	-4	-4

Table 3-2: Derivatives of interpolation functions, h_i

$$\begin{bmatrix} J_{11} \end{bmatrix}_{3x3} = \begin{bmatrix} X_{,r} & Y_{,r} & Z_{,r} \\ X_{,s} & Y_{,s} & Z_{,s} \\ X_{,t} & Y_{,t} & Z_{,t} \end{bmatrix} = \begin{bmatrix} x_2 - x_4 & y_2 - y_4 & z_2 - z_4 \\ x_3 - x_4 & y_3 - y_4 & z_3 - z_4 \\ x_1 - x_4 & y_1 - y_4 & z_1 - z_4 \end{bmatrix}$$

and

$$\begin{aligned} x_{,r} &= \sum_i h_{i,r}\ x_i = x_2 - x_4 \\ y_{,s} &= \sum_i h_{i,s}\ y_i = y_3 - y_4 \\ &\quad ,\ etc. \end{aligned}$$

$$\begin{bmatrix} J_{21} \end{bmatrix}_{6x3} = \begin{bmatrix} X_{,rr} & Y_{,rr} & Z_{,rr} \\ X_{,ss} & Y_{,ss} & Z_{,ss} \\ X_{,tt} & Y_{,tt} & Z_{,tt} \\ X_{,rs} & Y_{,rs} & Z_{,rs} \\ X_{,st} & Y_{,st} & Z_{,st} \\ X_{,rt} & Y_{,rt} & Z_{,rt} \end{bmatrix} = \begin{bmatrix} 0 \end{bmatrix}$$

$$\begin{aligned} x_{,rr} &= \sum_i h_{i,rr}\ x_i = 4(x_2 + x_4 - 2x_8) = 0 \\ y_{,ss} &= \sum_i h_{i,ss}\ y_i = 4(y_1 + y_3 - 2y_7) = 0 \\ &\quad ,\ etc. \end{aligned}$$

$$[J_{22}]_{6x6} = \begin{bmatrix} X_{,r}^{2} & Y_{,r}^{2} & Z_{,r}^{2} & 2X_{,r}Y_{,r} & 2Y_{,r}Z_{,r} & 2Z_{,r}X_{,r} \\ X_{,s}^{2} & Y_{,s}^{2} & Z_{,s}^{2} & 2X_{,s}Y_{,s} & 2Y_{,s}Z_{,s} & 2Z_{,s}X_{,s} \\ X_{,t}^{2} & Y_{,t}^{2} & Z_{,t}^{2} & 2X_{,t}Y_{,t} & 2Y_{,t}Z_{,t} & 2Z_{,t}X_{,t} \\ X_{,r}X_{,s} & Y_{,r}Y_{,s} & Z_{,r}Z_{,s} & X_{,r}Y_{,s}+X_{,s}Y_{,r} & Y_{,r}Z_{,s}+Y_{,s}Z_{,r} & X_{,r}Z_{,s}+X_{,s}Z_{,r} \\ X_{,s}X_{,t} & Y_{,s}Y_{,t} & Z_{,s}Z_{,t} & X_{,s}Y_{,t}+X_{,t}Y_{,s} & Y_{,s}Z_{,t}+Y_{,t}Z_{,s} & X_{,s}Z_{,t}+X_{,t}Z_{,s} \\ X_{,t}X_{,r} & Y_{,t}Y_{,r} & Z_{,t}Z_{,r} & X_{,r}Y_{,t}+X_{,t}Y_{,r} & Y_{,r}Z_{,t}+Y_{,t}Z_{,r} & X_{,r}Z_{,t}+X_{,t}Z_{,r} \end{bmatrix}$$

Therefore, the simplified relationships are obtained as

$$\begin{bmatrix} \frac{\partial^2}{\partial r^2} \\ \frac{\partial^2}{\partial s^2} \\ \frac{\partial^2}{\partial t^2} \\ \frac{\partial^2}{\partial r \partial s} \\ \frac{\partial^2}{\partial s \partial t} \\ \frac{\partial^2}{\partial r \partial t} \end{bmatrix} = \begin{bmatrix} \\ \\ & J_{22} = \text{const} & \\ \\ \\ \end{bmatrix} \begin{bmatrix} \frac{\partial^2}{\partial x^2} \\ \frac{\partial^2}{\partial y^2} \\ \frac{\partial^2}{\partial z^2} \\ \frac{\partial^2}{\partial x \partial y} \\ \frac{\partial^2}{\partial y \partial z} \\ \frac{\partial^2}{\partial z \partial x} \end{bmatrix} \qquad (3\text{–}26)$$

The second derivatives of the interpolation function $h_i(r,s,t)$ in Table 3-2 show that the second derivatives of the displacement, a, with respect to the isoparametric coordinates are constant throughout the element such as

$$\frac{\partial^2}{\partial r^2}a = \sum_i \frac{\partial^2 h_i}{\partial r^2}a_i = 4(a_1 + a_2 - 2a_5) = constant$$

$$and \quad etc.$$

Since the left-hand side of the equation in Table 3-2 is constant throughout the element, the second derivatives of the displacements with respect to the real coordinates are also constant throughout the element and can be obtained by using the Gauss elimination once for each element. Therefore, the residuals are constant throughout the element and can be computed exactly without using numerical integration.

Chapter 4

Sample Solutions

We present some examples that demonstrate the effective use of our algorithm for near-optimal mesh generation. There is no unique way in constructing the near-optimal meshes. If we have enough information about the problem already, we might be able to construct a near-optimal mesh within a given accuracy of solution in one step by controlling the key node density carefully on a loop-boundary. Otherwise, we may need to construct a coarse mesh and perform the adaptive refinement process several times. However, in general, the desirable way would be to construct an initial mesh as close as possible to a near-optimal mesh and limit the number of refinement iterations to about 2 to 3 iterations.

The following examples are presented in the order in which the basic algorithms are described in Chapters 2 and 3. Therefore, two dimensional problems using the triangular elements and the quadrilateral elements are considered first and then three dimensional problems are considered.

In the example (4-1), two different error indicators are employed for the refinement process in order to compare the efficiency of the error indicators : one uses the energy norm without a relaxation factor as in equation (2-24), and the other uses the energy norm with a relaxation factor as in equation (2-25). In all the other examples, the error indicator with the relaxation factor is used for the error analysis.

4.1 Two-Dimensional Application

4.1.1 Examples Using Triangular Elements

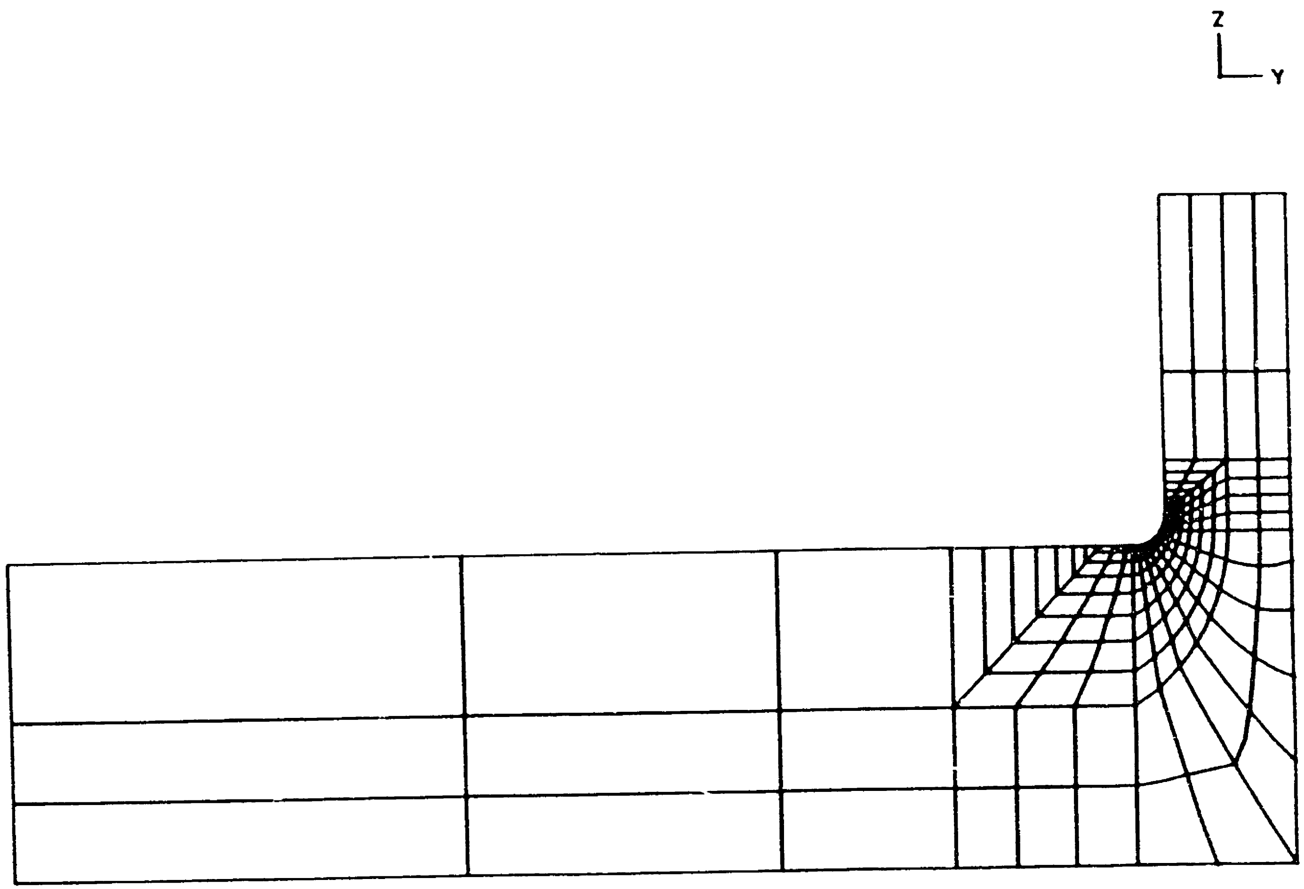
Three examples are considered for the use of the 6-node triangular element. This element is equivalent to the degenerated 8-node quadrilateral element with the isotropic correction in ADINA [50]. The examples are chosen to solve a problem in axisymmetric, plane stress and plane strain conditions, respectively.

(Example 4-1) Analysis of an Axisymmetric Pressure Vessel

The first example concerns the analysis of the axisymmetric pressure vessel shown in Figure 2-21(a). The goal of the analysis is to determine the stress distribution due to the internal pressure loading on the "line of stress output" shown in Figure 2-21(b).

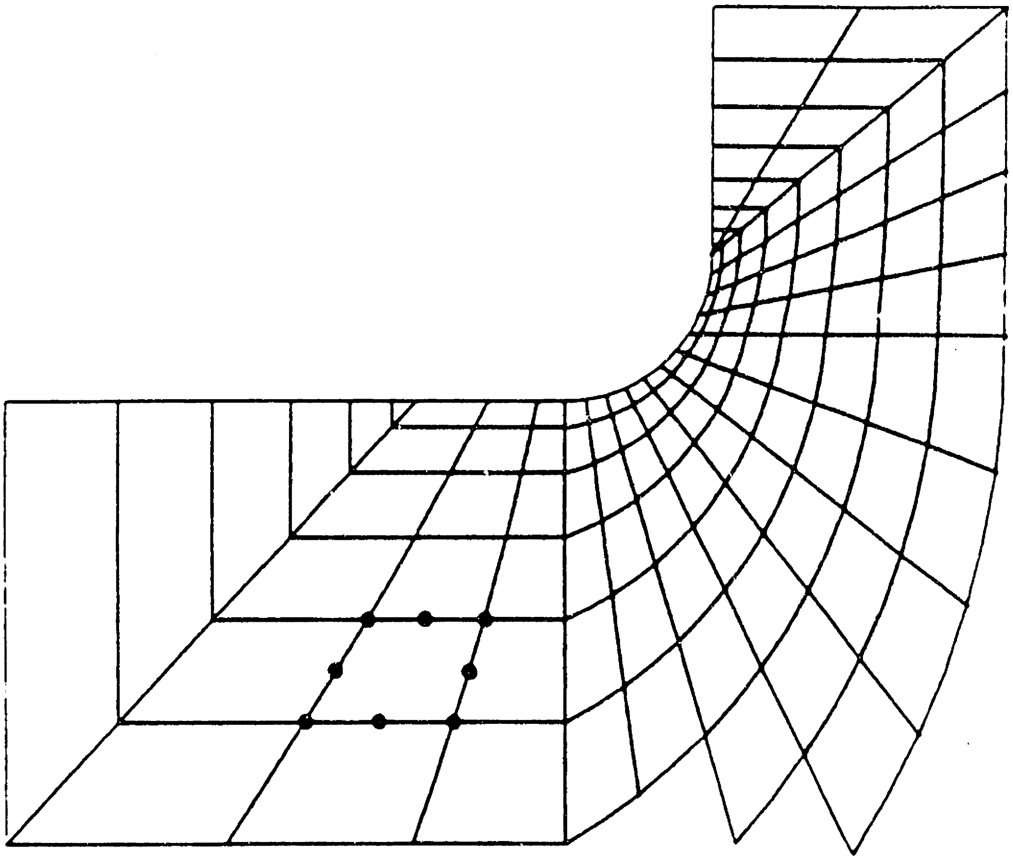
This problem was first considered in a paper by Floyd [59] and more rigorous analyses have been performed by Sussman and Bathe [60-61]. In their paper, Sussman and Bathe have constructed a near-optimal mesh with 181 eight-node elements to obtain a resonable accuracy of solution. The mesh constructed is shown in Figure 4-1 and the graphs of the principal stresses along the line of interest are shown in Figure 4.2. These results compare farely well with the photoelastic results [59].

Two different error indicators are used to compare the efficiency of these indicators in the refinement process. For this purpose an initial coarse mesh is constructed using the program AMESH-I. In order to construct an initial mesh



(a) Complete mesh

Figure 4-1: 181 eight-node element mesh for axisymmetric pressure vessel [61]



(b) De'ail of mesh

Figure 4-1: continued

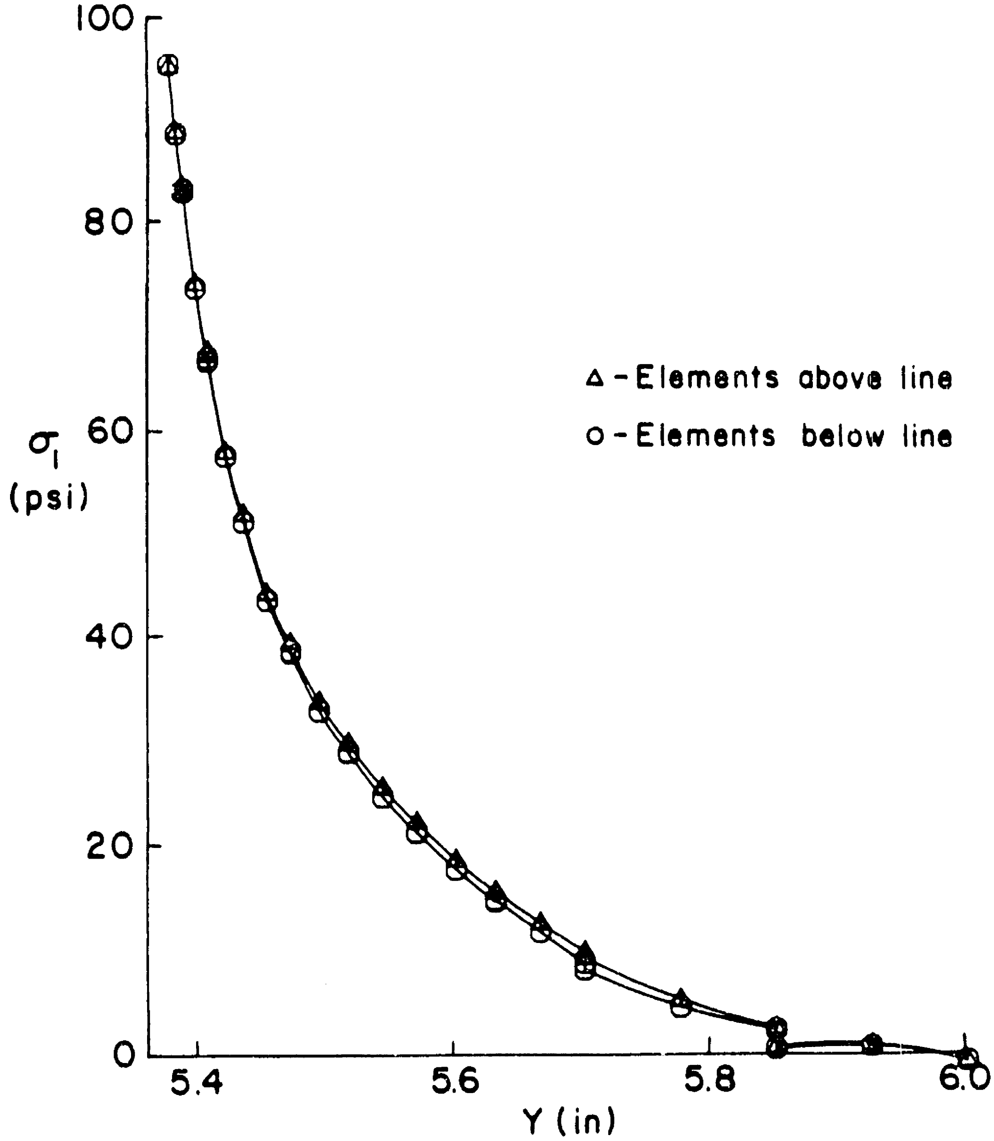


Figure 4-2 a): Maximum in-plane principal stress for 181 element mesh [61]

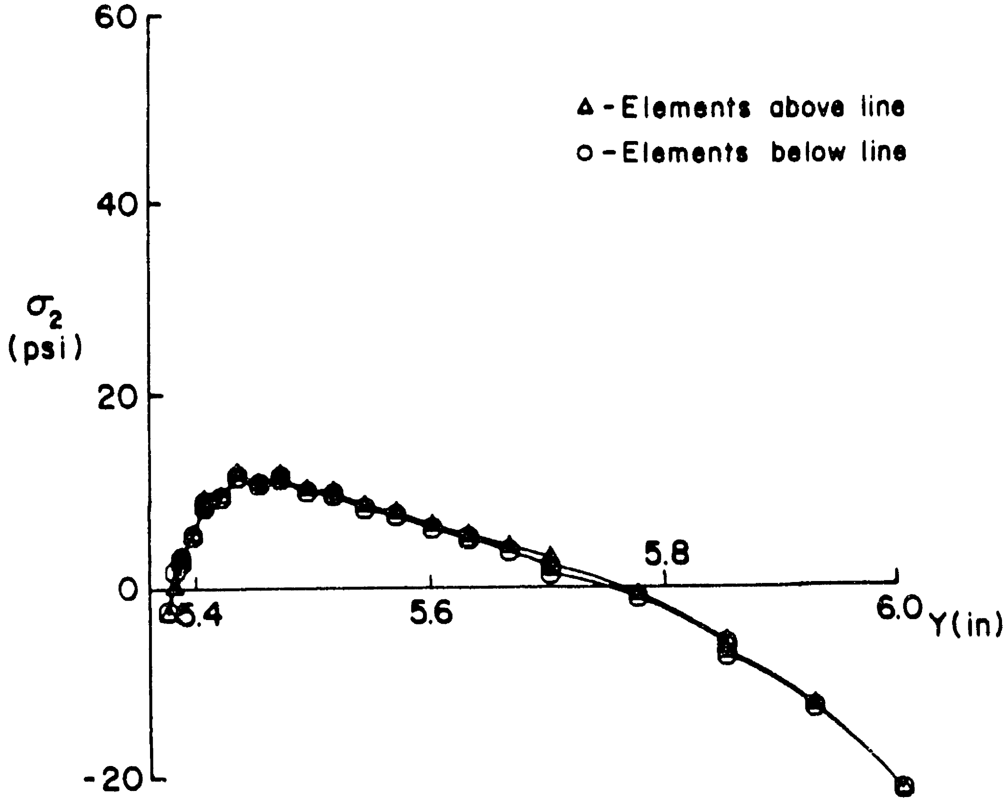


Figure 4-2 b): Minimum in-plane pricipal stress for 181 element mesh [61] with well conditioned elements, the analysis domain is subdivided into four near-convex subdomains as shown in Figure 4-3.

The triangulation algorithm automatically constructed the 104 triangular element mesh shown in Figure 4-4. For this mesh, the stresses are computed at the nodal points and the principal stress distributions on the "line of stress output" are shown in Figure 4.5.

The figure shows the stresses calculated from the elements above the line of interest and from the elements below that line. The results in Figure 4-5 show large stress discontinuities between adjacent elements, which means that the mesh is not satisfactory.

If we use the error indicator with the relaxation factor, *1/h*, (see equation (2-25)) for the adaptive refinement process, a 281 element mesh is obtained in 2 steps (104 element → 236 element → 281 element) within a given error tolerance ($\varepsilon_{tolerance} = 0.02$), see the Figure 4-6. The stress output results for the 281 element mesh are shown in Figure 4-7. These reults compare favorably with the results by Sussman and Bathe and other experimental results [59-61].

Here the error indicator used does not have a direct relationship with the actual stress error involved. If we use the error indicator in the energy norm as in equation (2-24) with $\varepsilon^*_{tolerance} = 0.0004$ for the adaptive refinement process, a 278 element mesh is obtained in 2 steps (104 elements → 266 elements → 278 elements), see Figure 4-8. In this case, the error tolerance is chosen ($\varepsilon^*_{tolerance} = 0.0004$) to generate a similar number of elements (281 elements) after 2 steps of refinement in order to compare the efficiencies of the two different error indicators.

The stress output results for this analysis are shown in Figure 4-9,

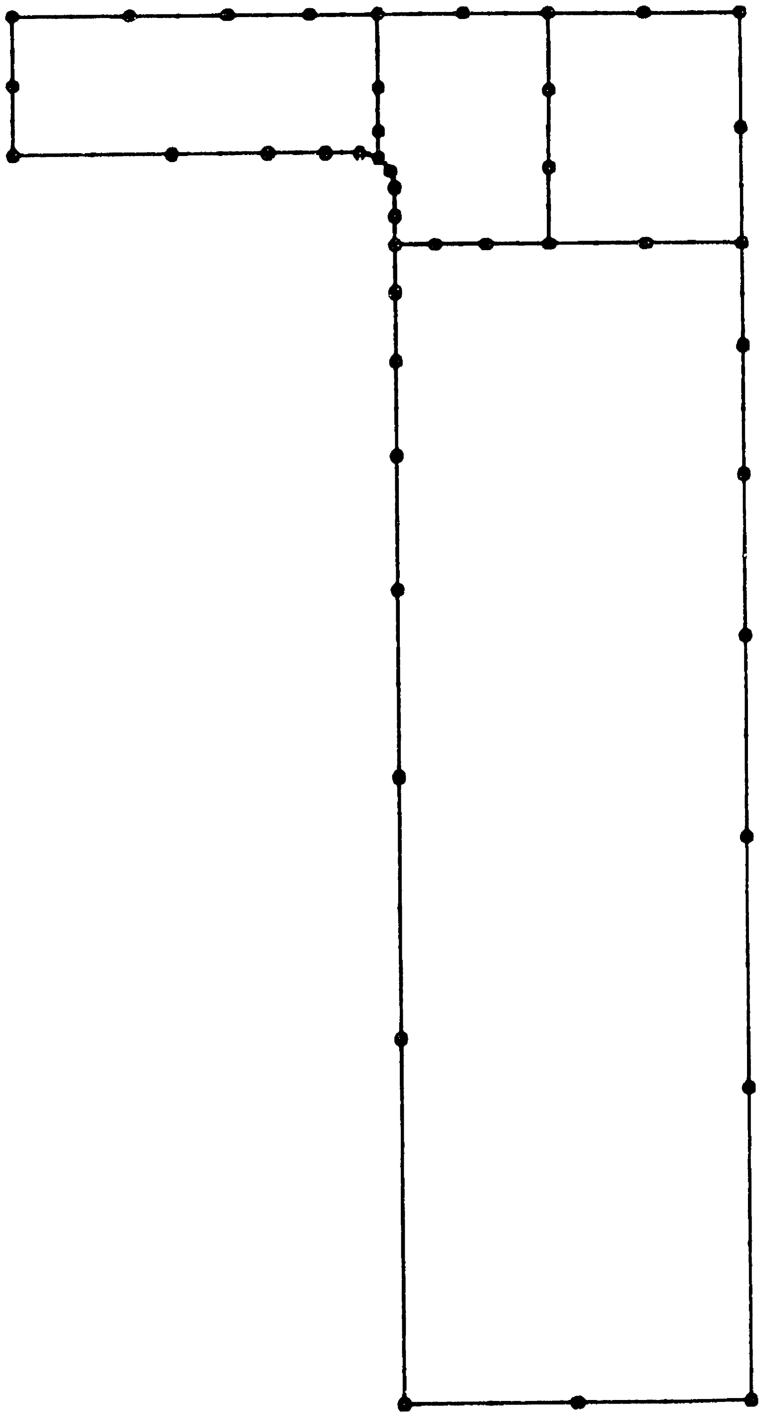
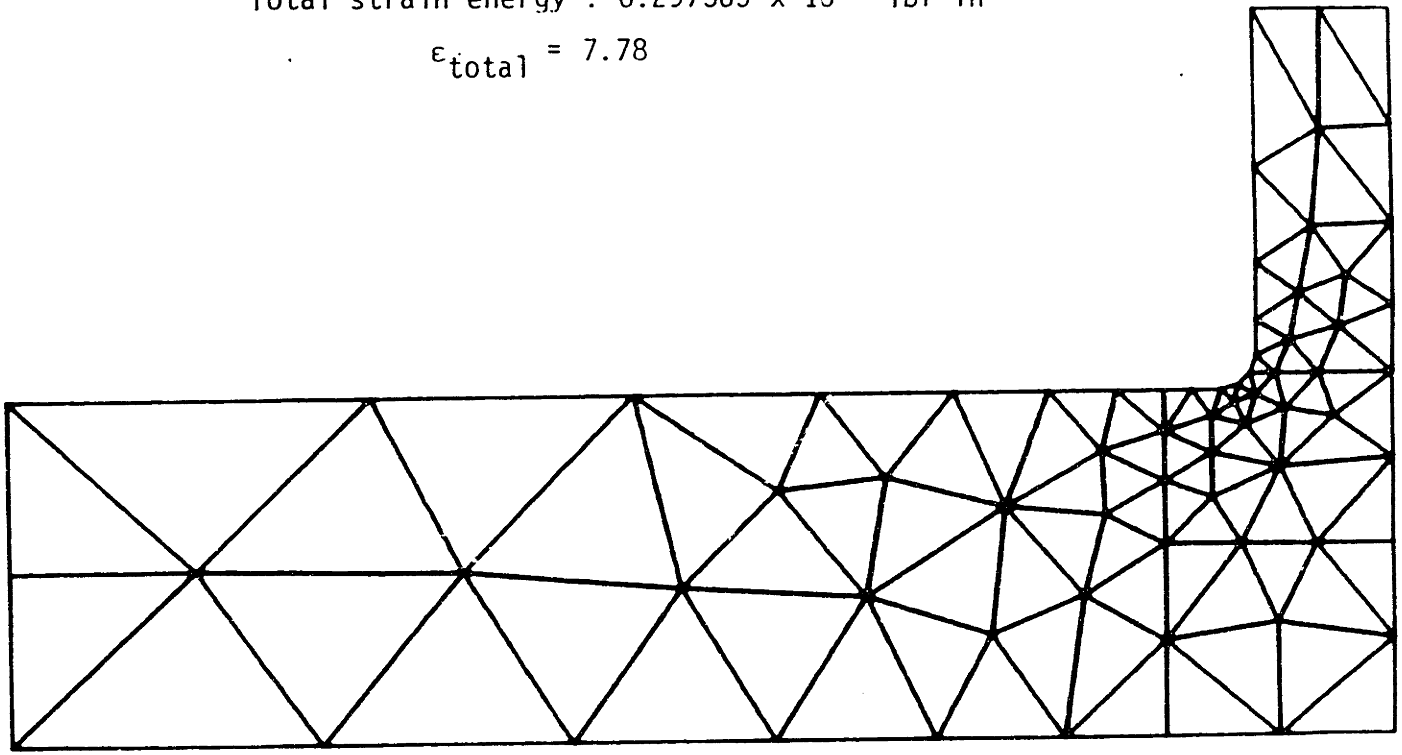


Figure 4-3: Subdivided domains with key nodes



Total strain energy : 0.297365 x 10^{-2} lbf-in

$\varepsilon_{total} = 7.78$

Figure 4-4: Coarse starting mesh with 104 elements

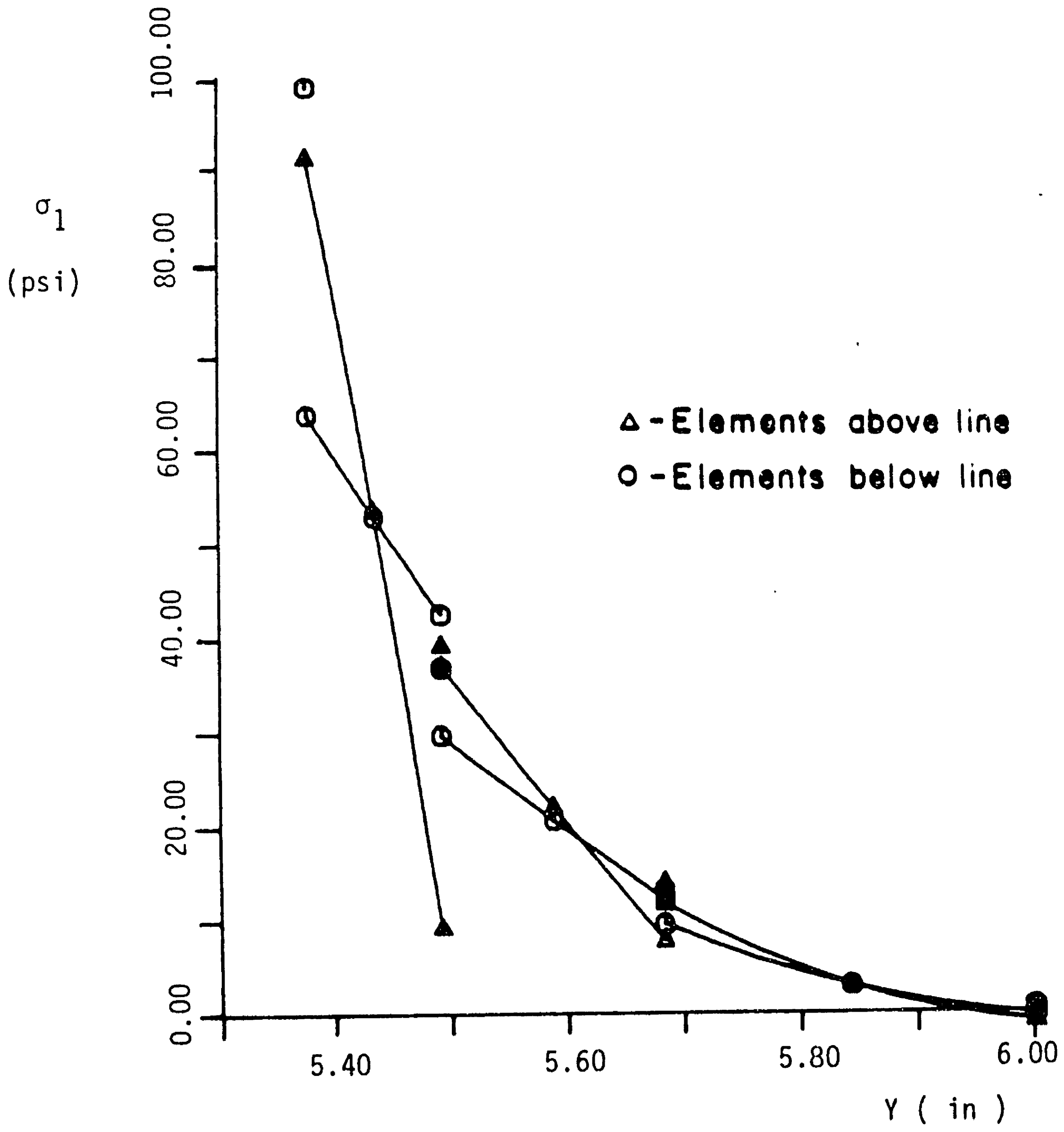


Figure 4-5 a): Maximum in-plane principal stress for 104 element mesh

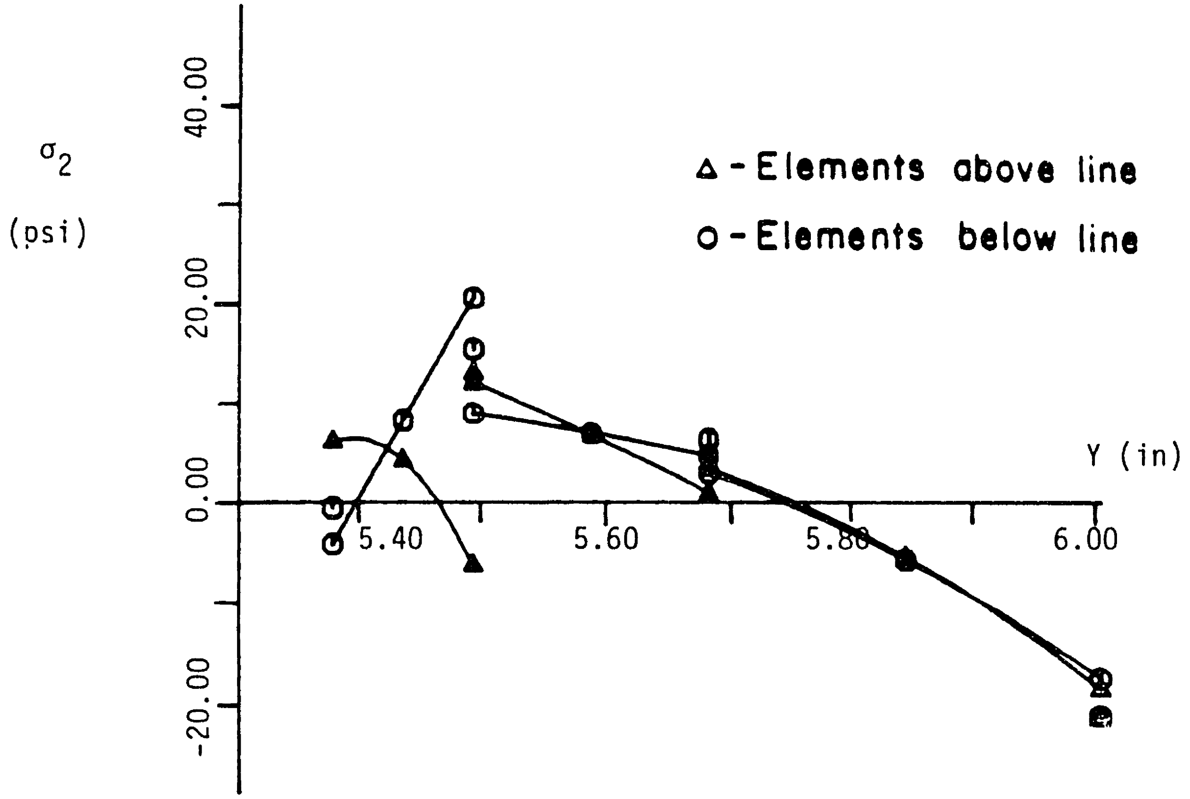


Figure 4-5 b): Minimum in-plane principal stress for 104 element mesh

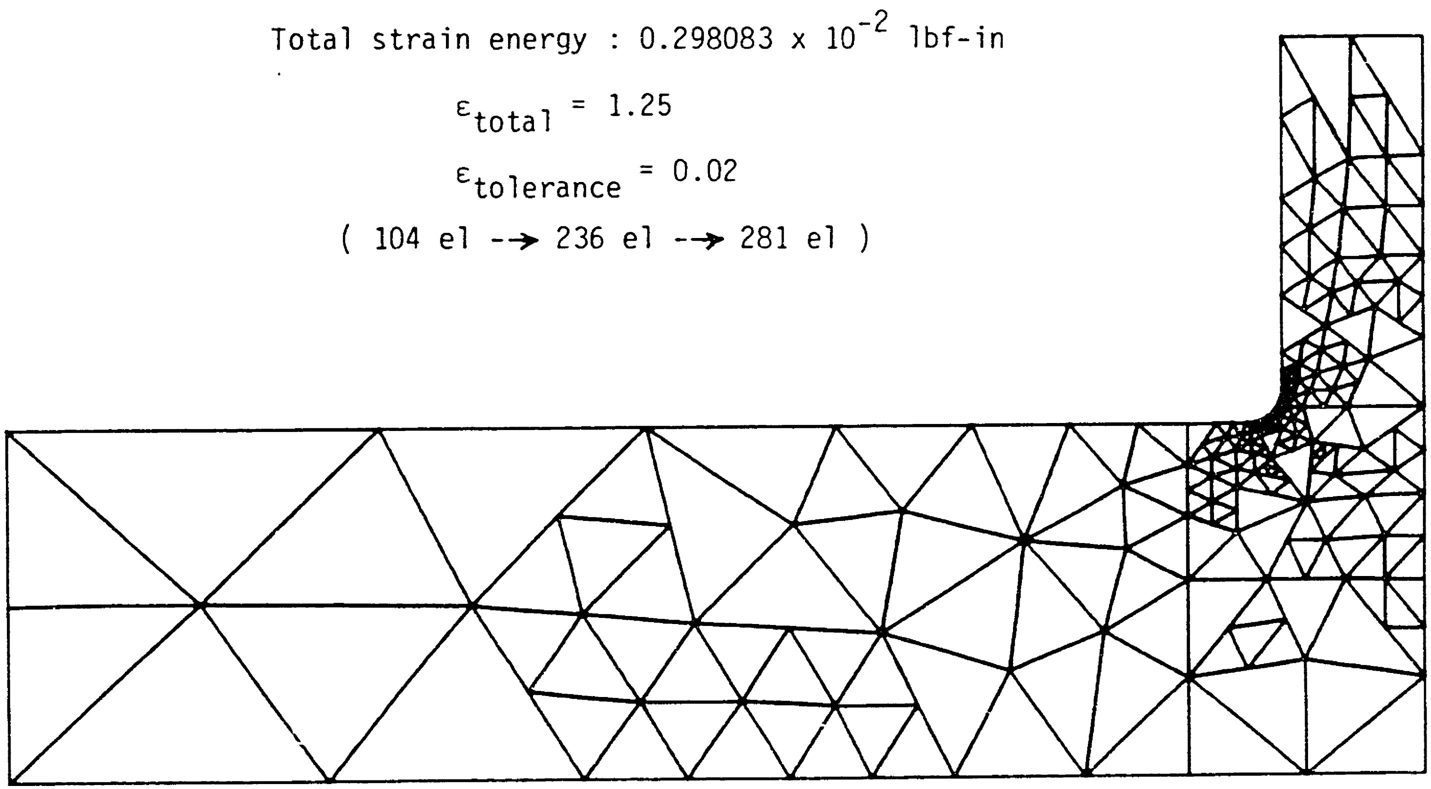

Total strain energy : 0.298083 x 10^{-2} lbf-in
ε_{total} = 1.25
$\varepsilon_{tolerance}$ = 0.02
(104 el --> 236 el --> 281 el)

Figure 4-6 a): Final mesh with 281 elements using the error indicator with a relaxation factor

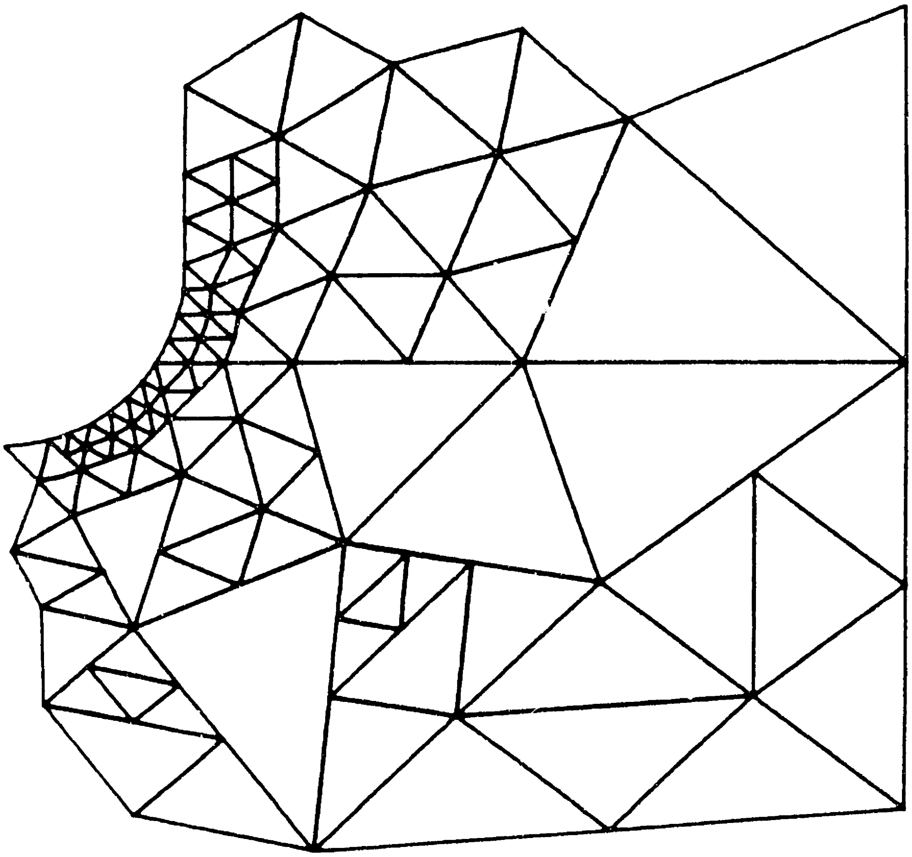


Figure 4-6 b): Detail of 281 element mesh

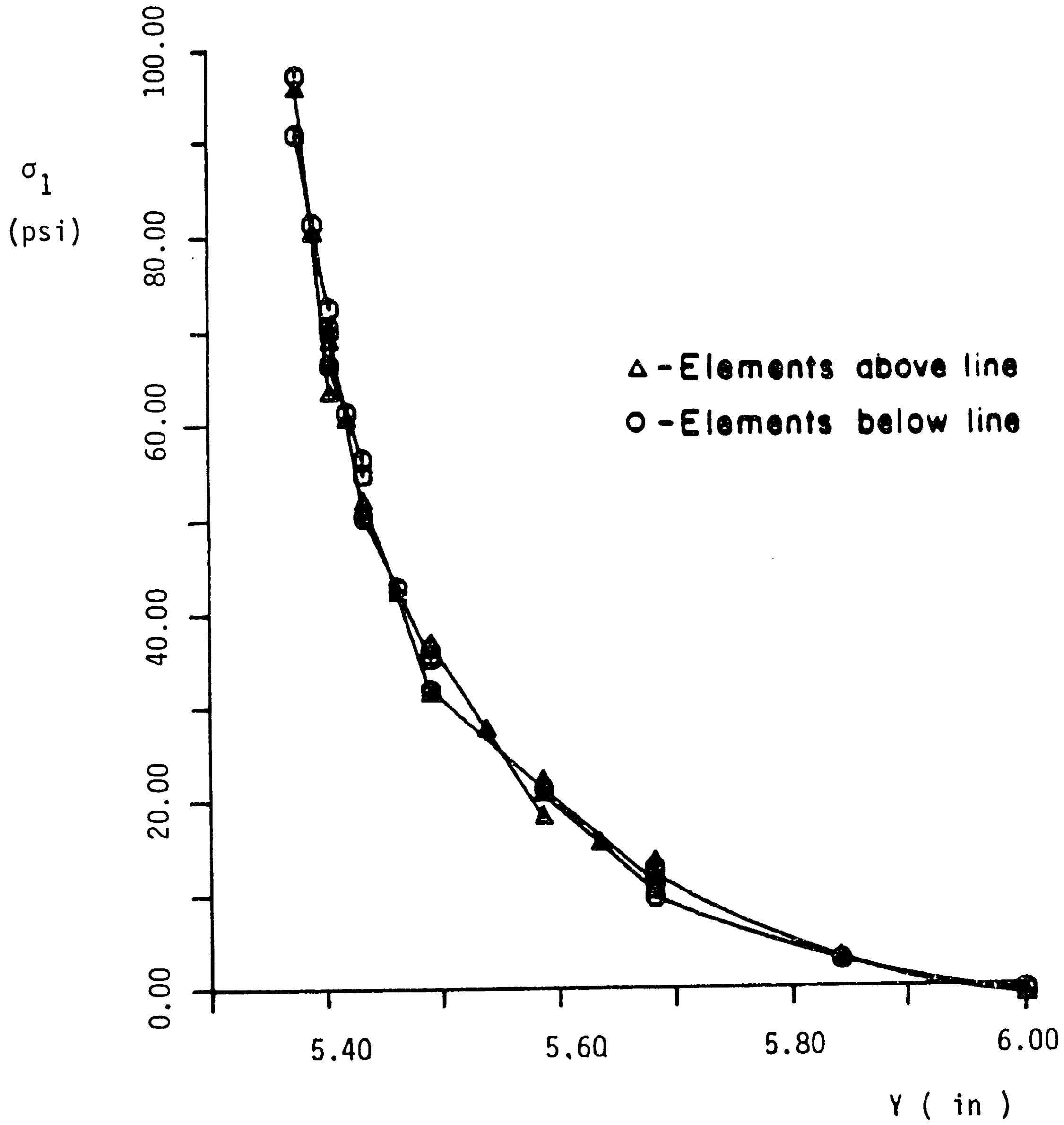


Figure 4-7 a): Maximum in-plane principal stress for 281 element mesh

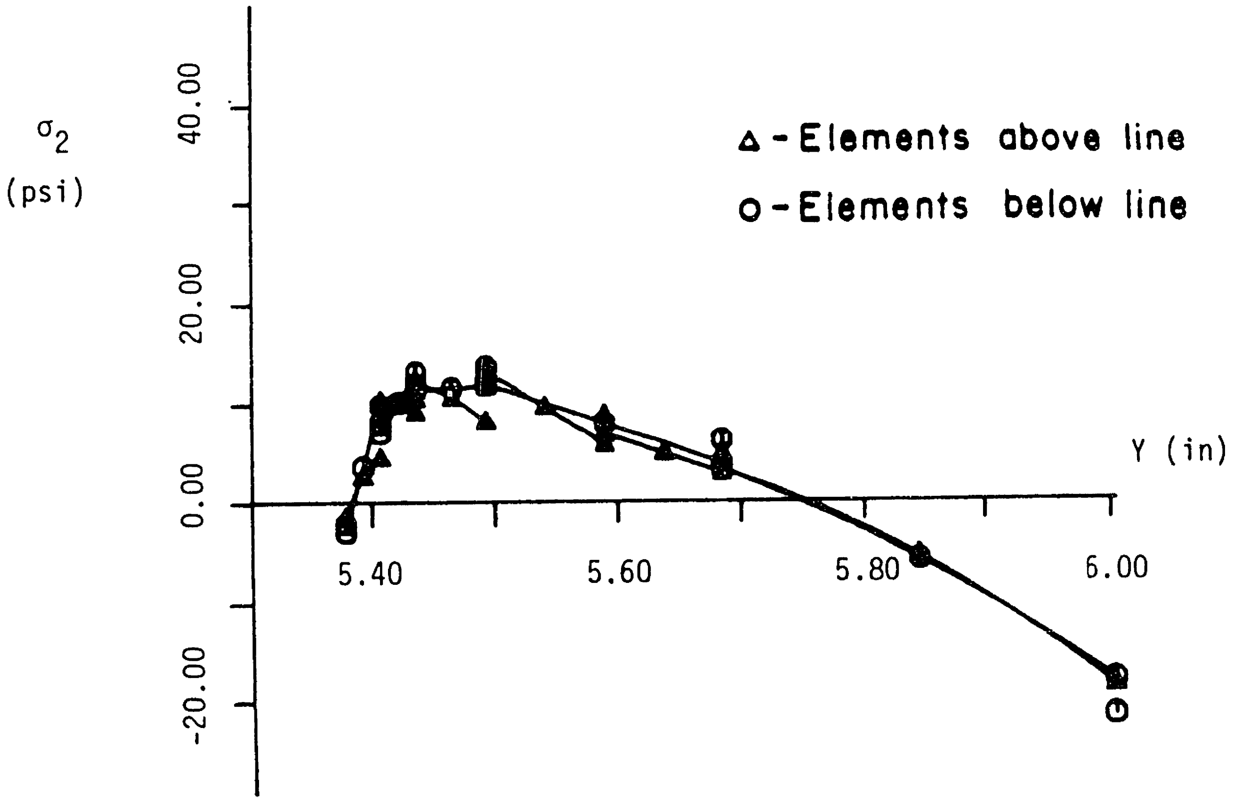
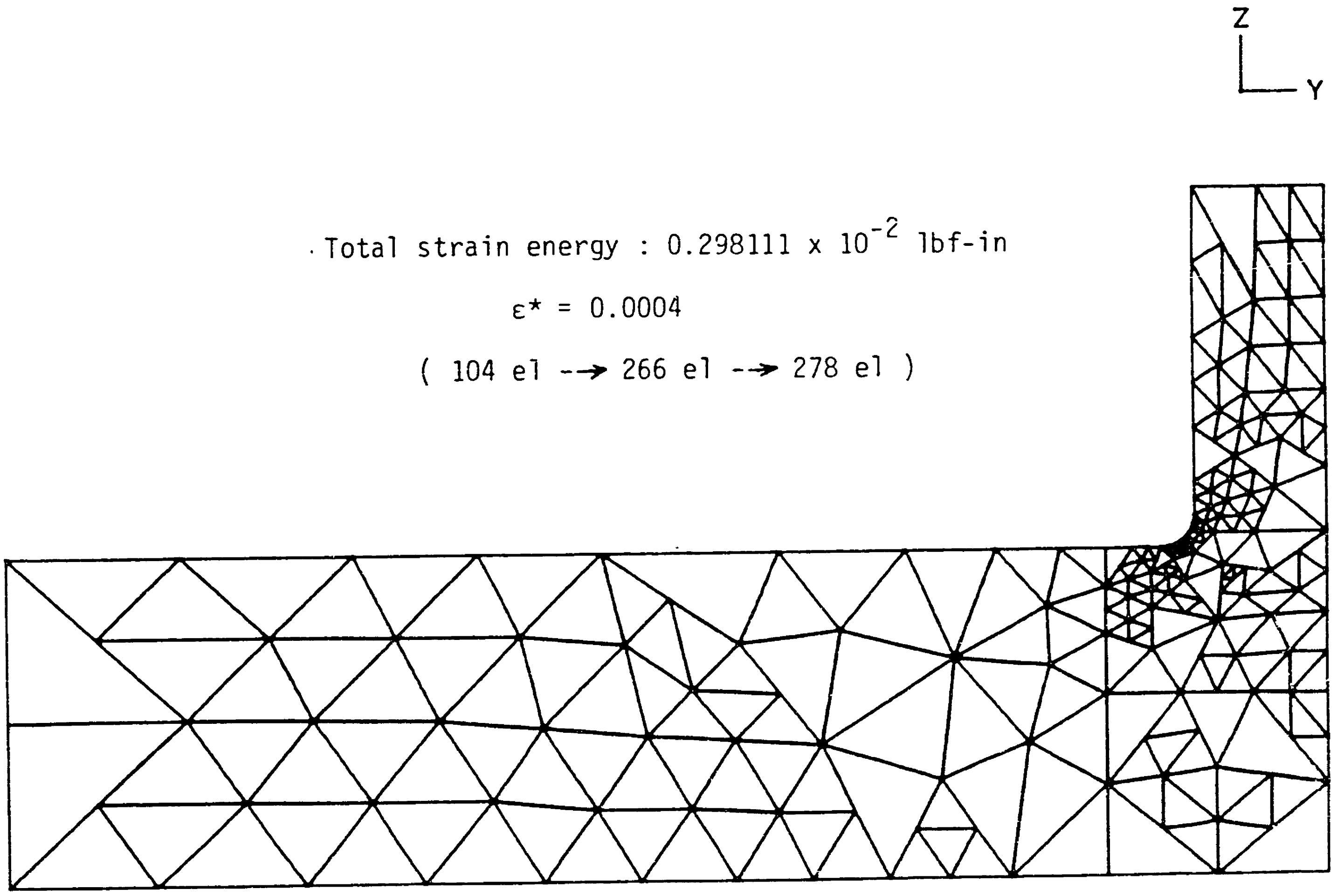


Figure 4-7 b): Minimum in-plane principal stress for 281 element mesh



Total strain energy : 0.298111 x 10^{-2} lbf-in

$\varepsilon^* = 0.0004$

(104 el --> 266 el --> 278 el)

Figure 4-8 a): Final mesh with 278 elementsn using the error indicator without a relaxation factor

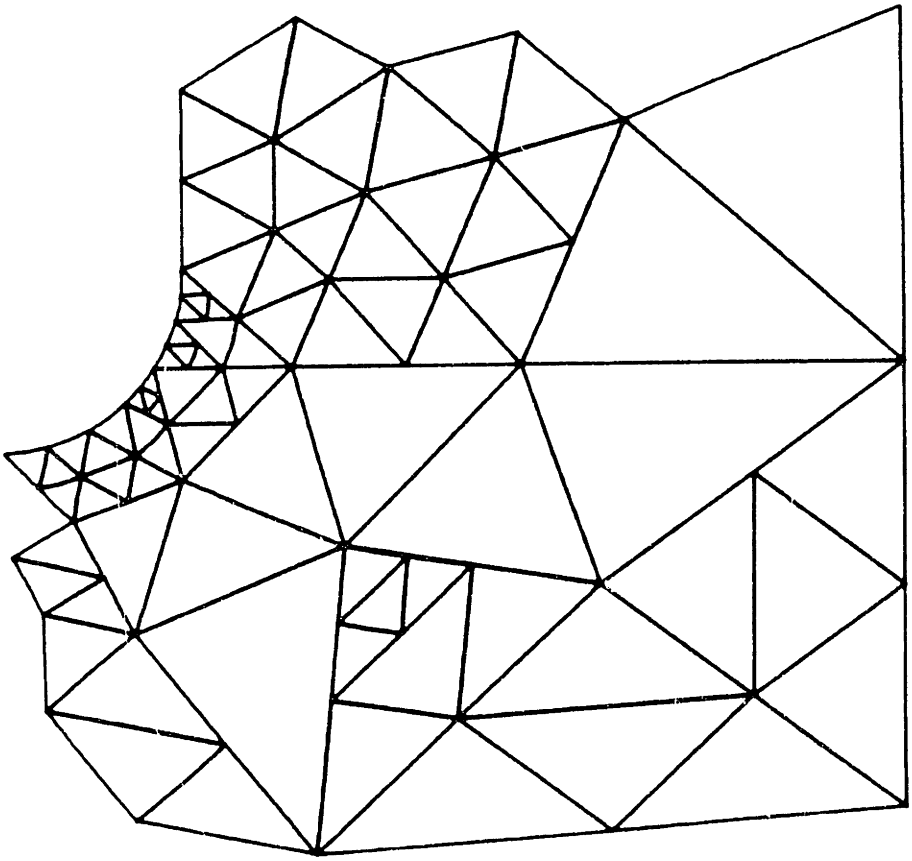


Figure 4-8 b): Detail of 278 element mesh

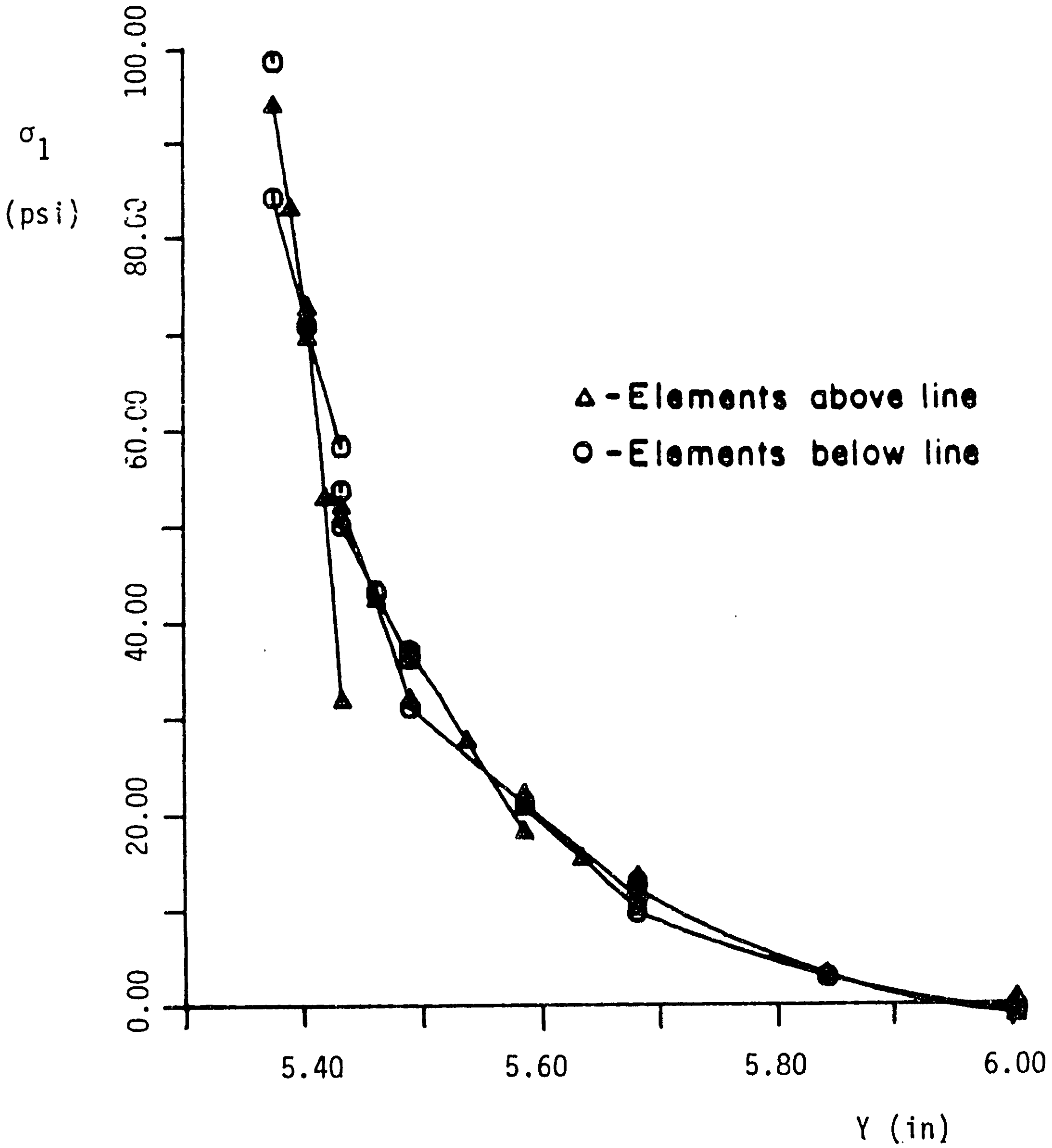


Figure 4-9 a): Maximum in-plane principal stress for 278 element mesh

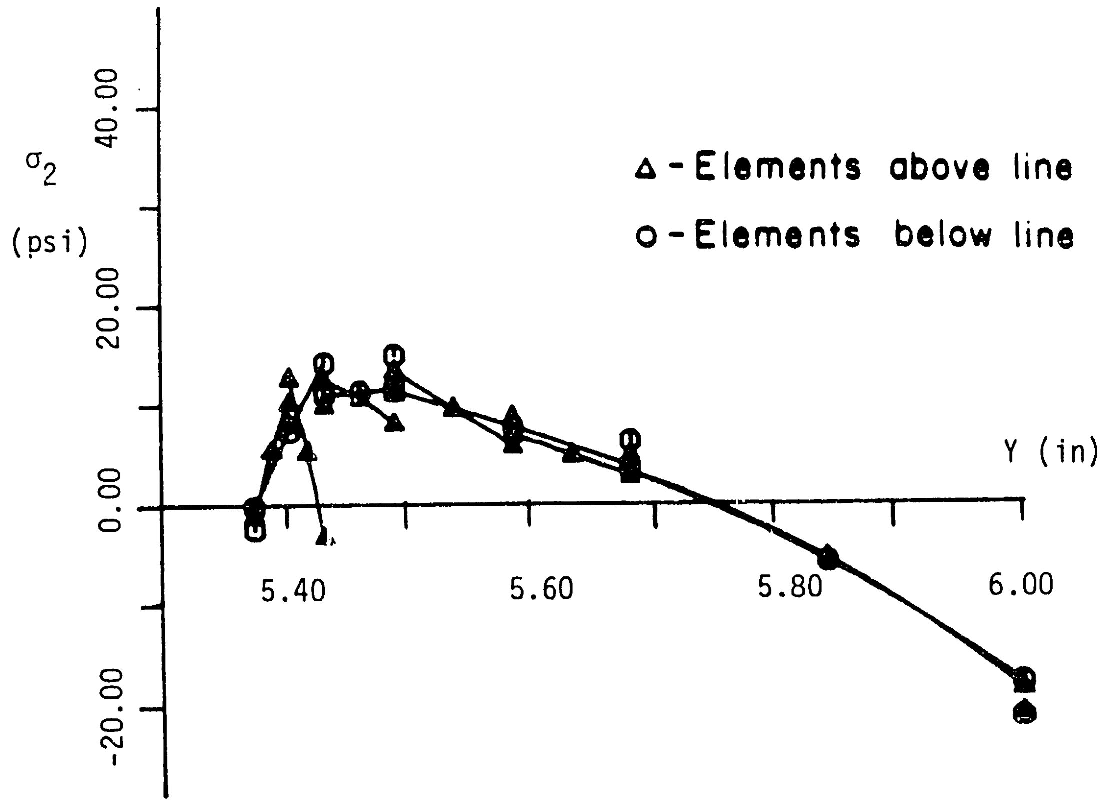


Figure 4-9 b): Minimum in-plane principal stress for 278 element mesh

which are not as satisfactory as the ones obtained for the 281 element mesh in Figure 4-7.

By compaiing the 281 element mesh in Figure 4-6 and the 278 element mesh , we can see that in the 281 element mesh, the refinement is more concentrated at the stress concentration. However, the total strain energy in the 281 element mesh is less than that of the 278 element mesh.

The reason is due to the relaxation factor, $1/h$, employed in the error indicator in equation (2-25). This relaxation factor is employed so that the refinement is more concentrated in the region of the small size elements, which results in better solutions at the stress concentration with little sacrifice on the overall accuracy. Therefore, from now on, we will use the error indicator with the relaxation factor in equation (2-25) for the adaptive refinement process.

If we have enough information about the problem already, such as the distribution of the error indicator throughout the mesh, or the region and the strength of the stress concentration, we might be able to reduce the number of refinement iterations by starting with a fine initial mesh.

Using the experiences obtained from the above analysis, a fine initial mesh is constructed by 221 elements as shown in Figure 4-10. For this 221 element mesh, the adaptive refinement is performed with the same error tolerance ($\varepsilon_{tolerance}$ = 0.02) and a 281 element mesh is obtained in one step, see the Figure 4-11. The stress output results for the 281 element mesh in Figure 4.12 show that better results can be obtained in only one step of refinement, if previous experiences are used in constructing a fine initial mesh. Therfore, the 281 element mesh in Figure 4-11 can bε considered as a near-optimal mesh to this problem for the error tolerance, $\varepsilon_{tolerance}$ = 0.02.

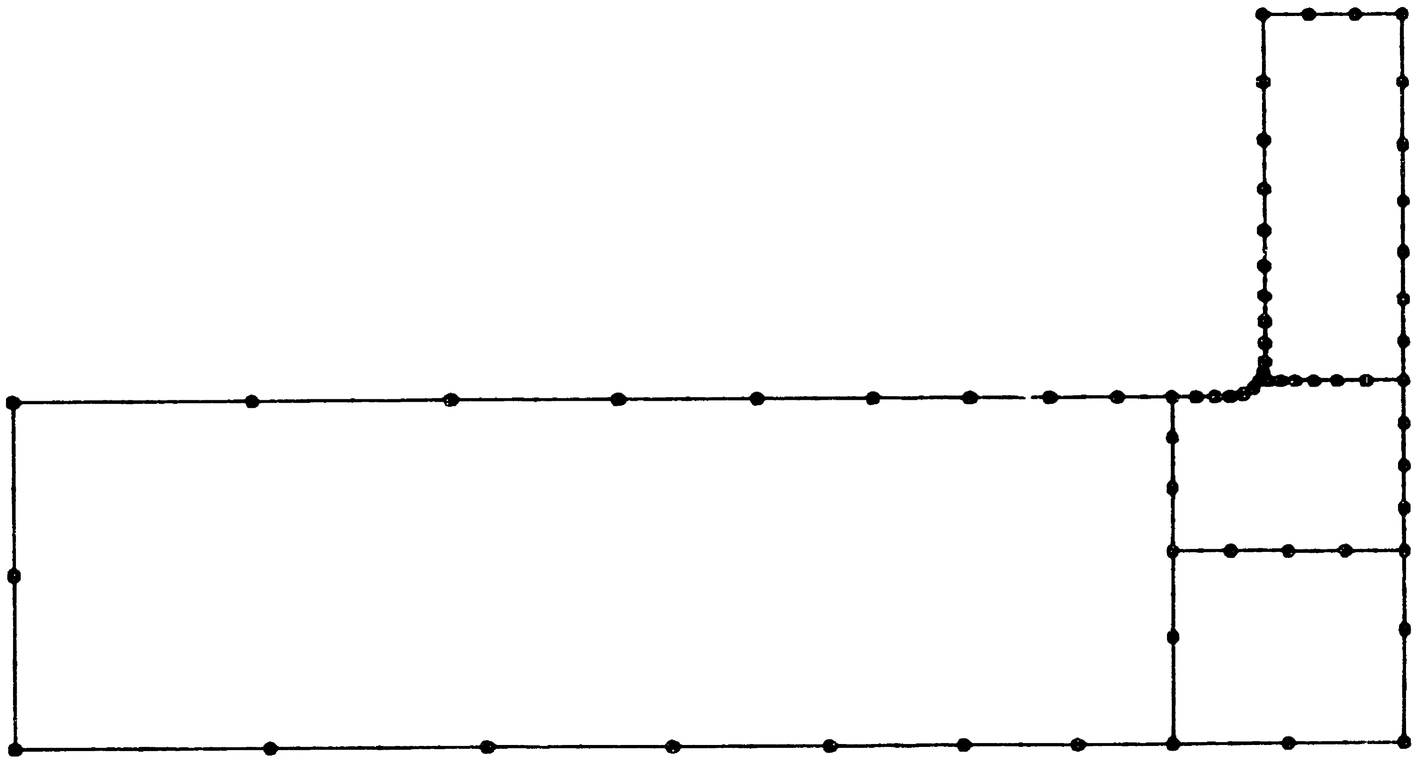
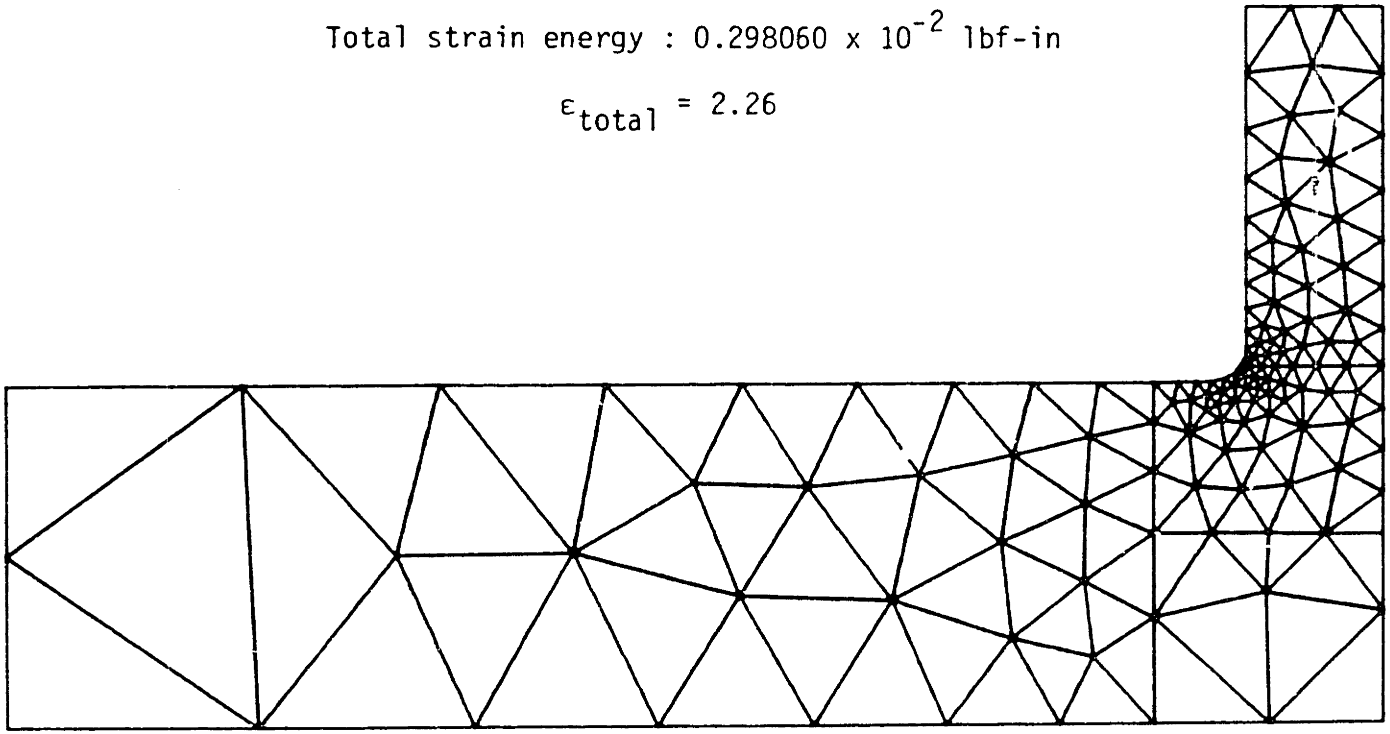


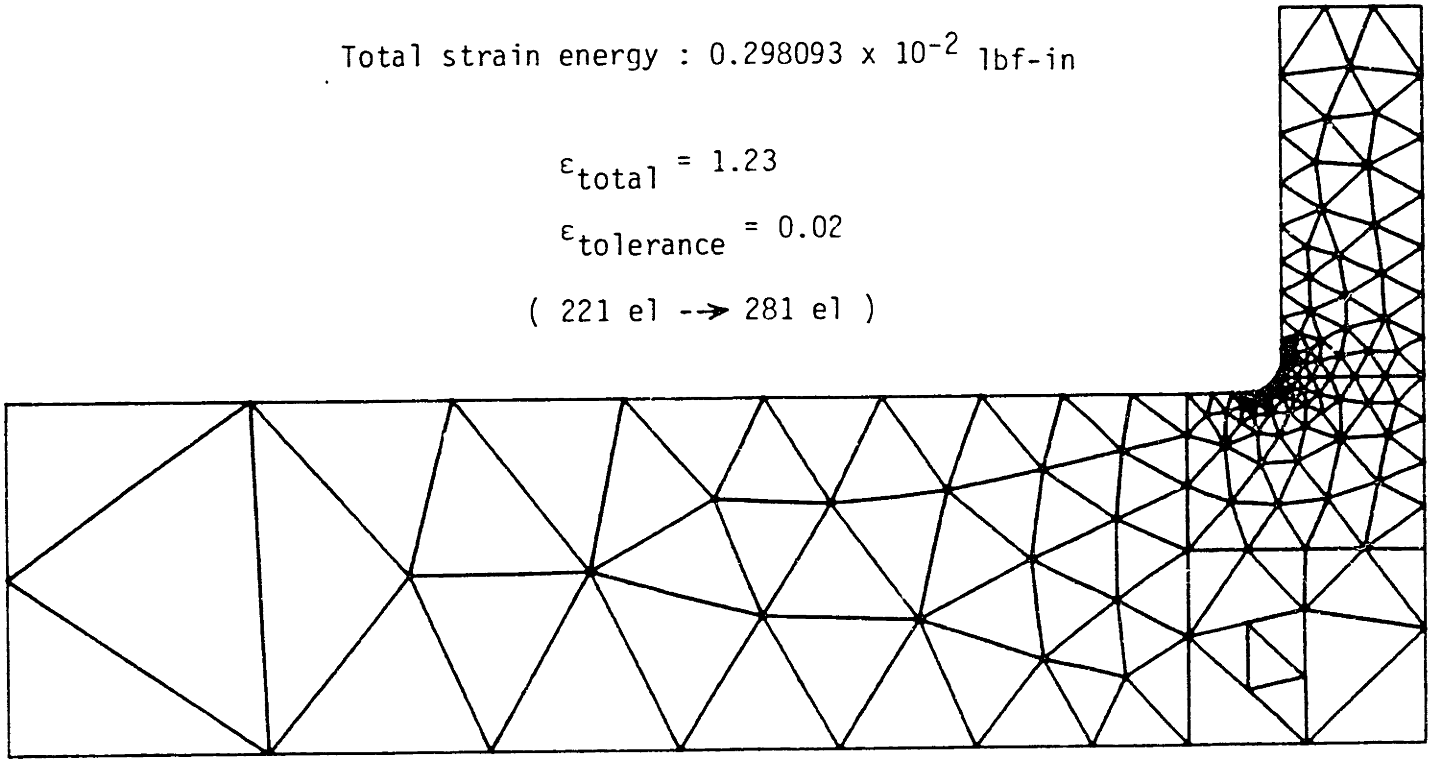
Figure 4-10 a): Key nodes distribution for a fine mesh construction



Total strain energy : 0.298060 x 10^{-2} lbf-in

$\varepsilon_{total} = 2.26$

Figure 4-10 b): A fine starting mesh with 221 elements



Total strain energy : 0.298093 x 10^{-2} lbf-in

ε_{total} = 1.23

$\varepsilon_{tolerance}$ = 0.02

(221 el --> 281 el)

Figure 4-11 a): Final 281 element mesh

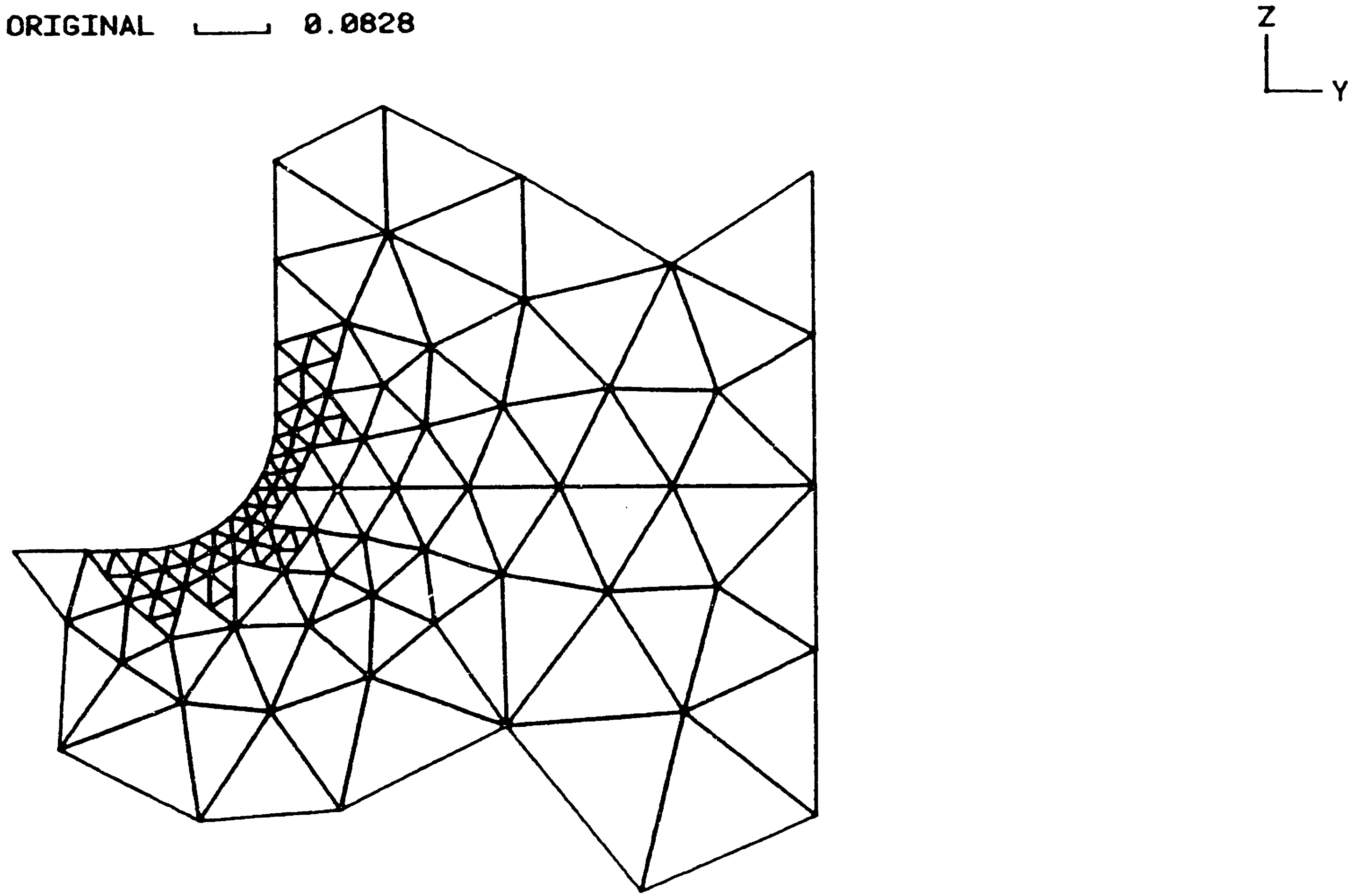


Figure 4-11 b): Detail of 281 element mesh

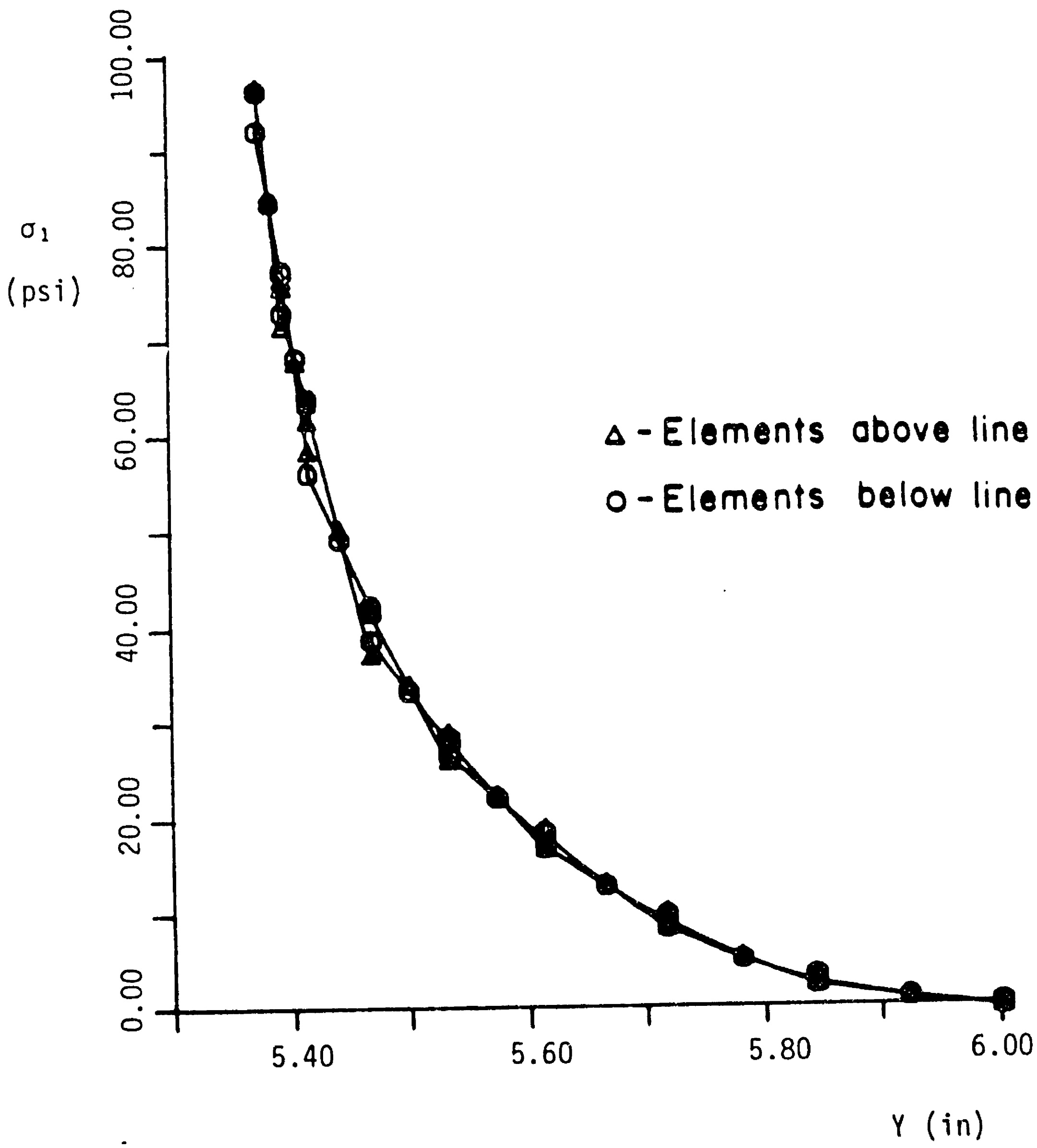


Figure 4-12 a): Maximum in-plane principal stress for 281 element mesh

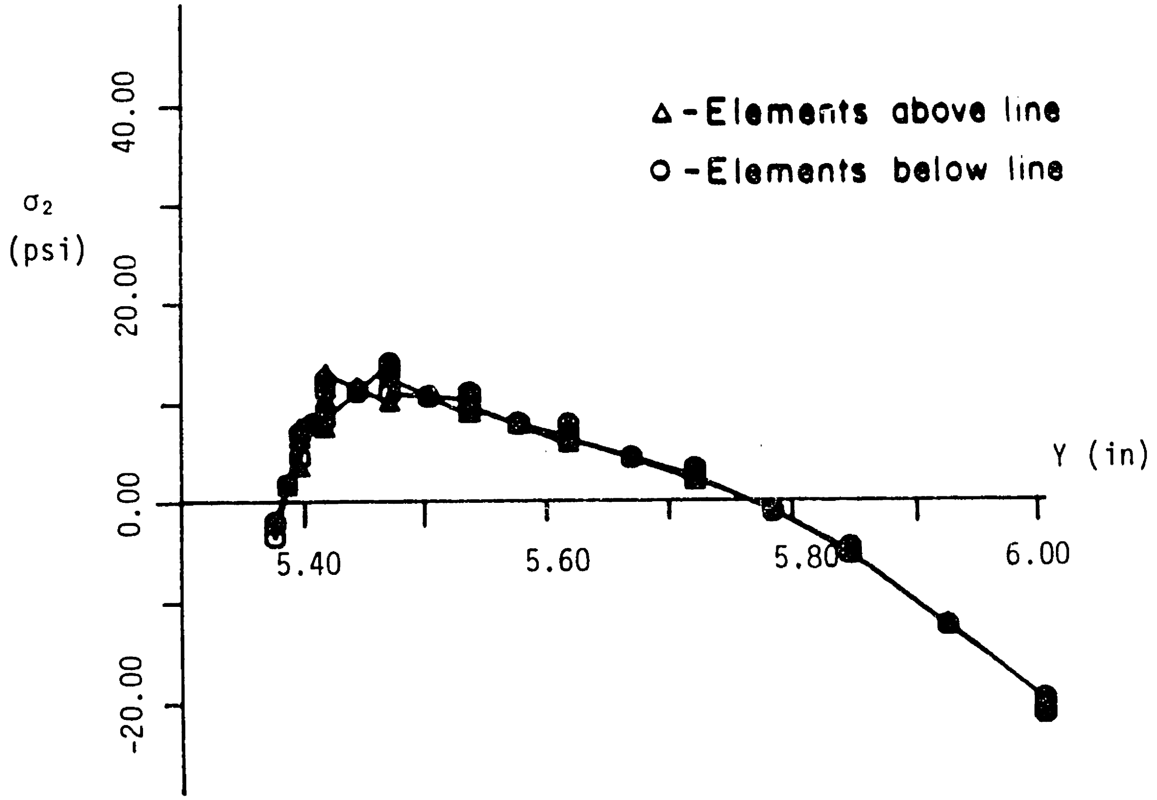


Figure 4-12 b): Minimum in-plane principal stress for 281 element mesh

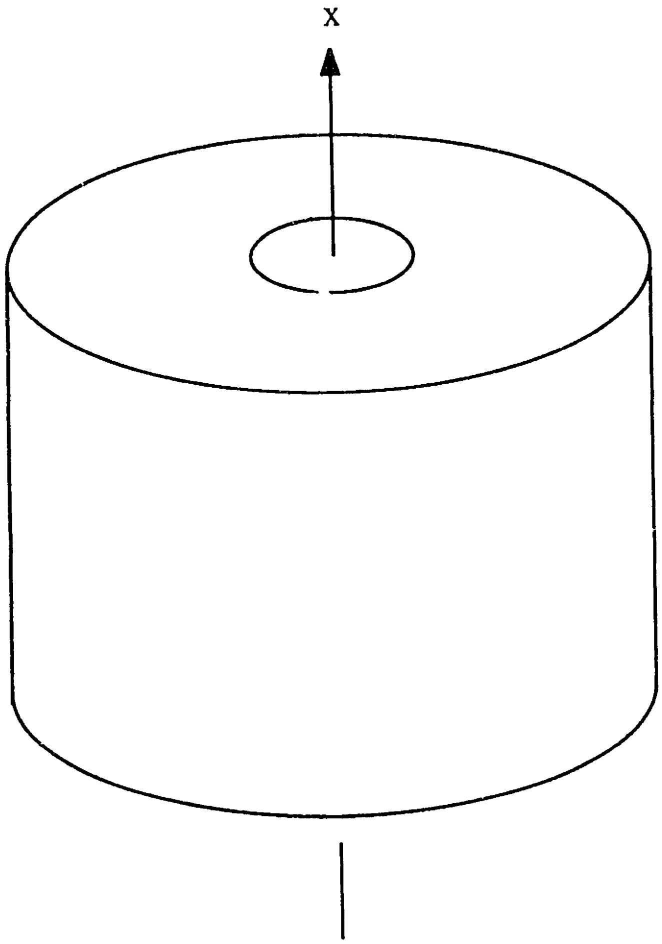
(Example 4-2) Analysis of a Thick Circular Cylinder under Internal Pressure (Plane Strain)

As a plane strain problem, we consider a thick cylinder under internal pressure as shown in Figure 4-13. The goal of this analysis is to determine the stress distribution along the line A-A'. The analytical solution for this problem is available in reference [62].

Two different initial meshes are constructed to get a near-optimal mesh, one coarse mesh and one fine mesh. Using the symmetry condition, only one quarter of a cylinder is considered. The coarse initial mesh with 17 elements is shown in Figure 4.14. In this coarse mesh, all the internal nodes are relocated after triangulation to keep the symmetry in order to see whether the adaptive refinement process gives symmetric results. Namely, our triangulation scheme can usually construct an almost symmetric mesh if the initial loop-boundary is symmetric, but not an exactly symmetric mesh. The reason is due to the directionality in our triangulation scheme described in Chapter.2.

The minimum principal stress output for this 17 element mesh along the line A-A' is shown in Figure 4-15, which does not compare well with the analytical solution.

To obtain good results, the error tolerance is set to $\varepsilon_{\text{tolerance}} = 0.002$ and the resulting 227 element mesh by the adaptive refinement process is obtained in 3 steps (17 elements → 68 elements → 155 elements → 227 elements), see Figure 4-16. As shown in Figure 4.16, the resulting mesh is also exactly symmetric. The σ_{yy} distribution along the line A-A' is shown in Figure 4.17, which compares well with the analytical solution.



Inner radius = 0.5 m
Outer radius = 2 m
Internal pressure
= 100 MPa

$E = 2.07 \times 10^5$ MPa
$\nu = 0.3$

Plane strain conditions in x-direction

Figure 4-13: Thick cylinder under internal pressure

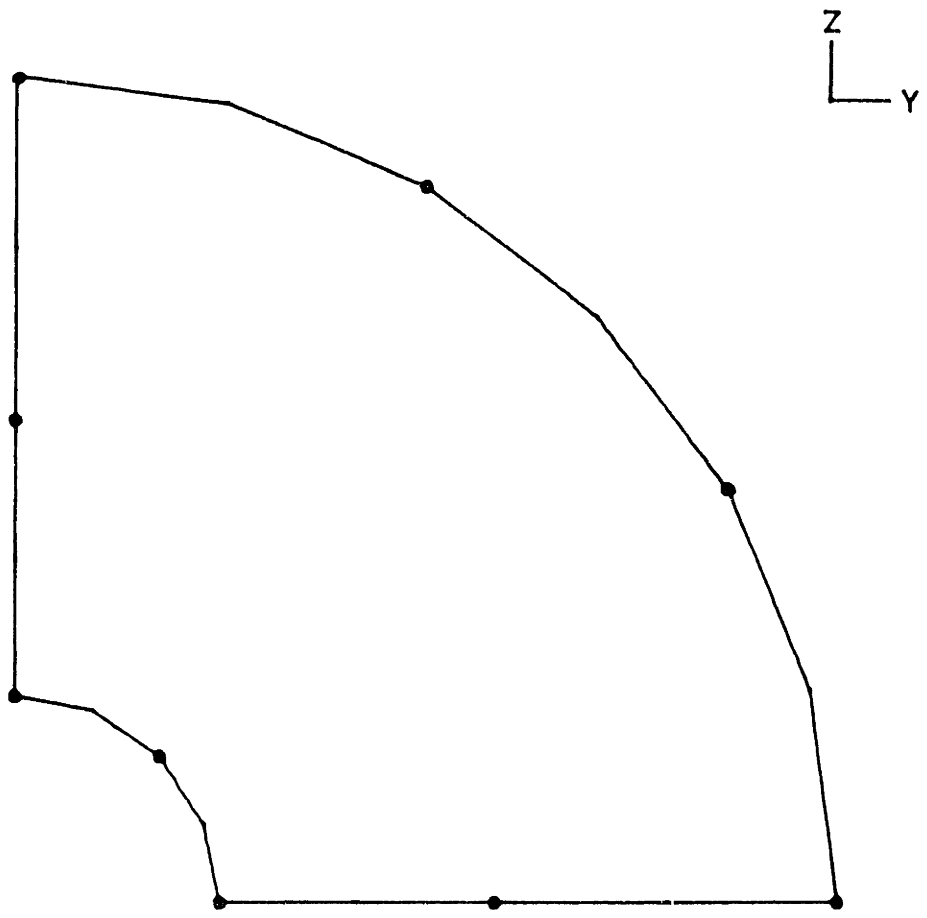
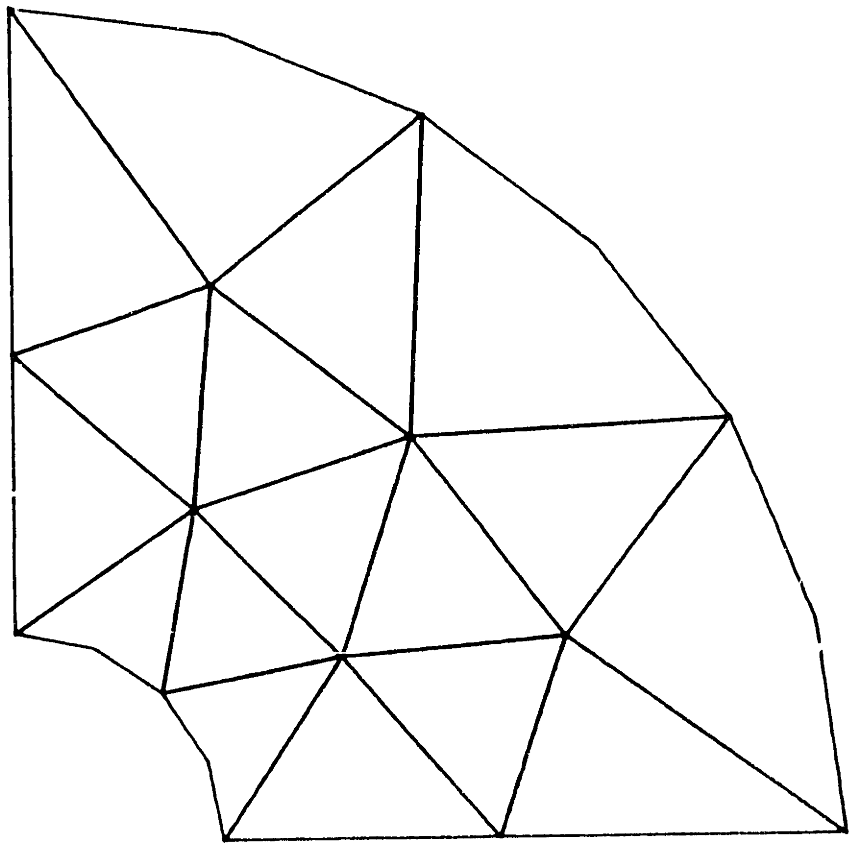


Figure 4-14 a): key nodes distribution for a coarse mesh generation



Total strain energy : 0.132614 x 10^5 N-m

$\varepsilon_{total} = 6.22$

Figure 4-14 b): Coarse starting mesh with 17 elements

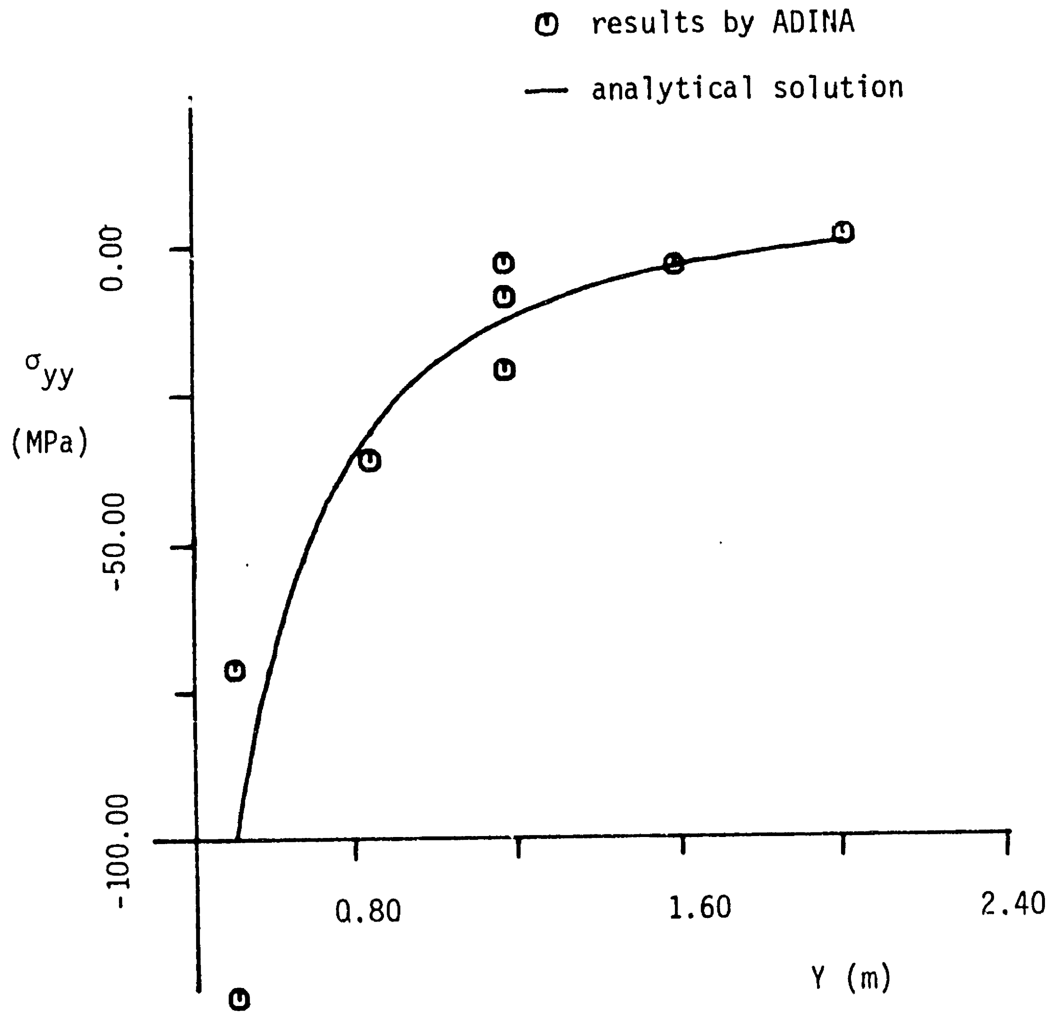
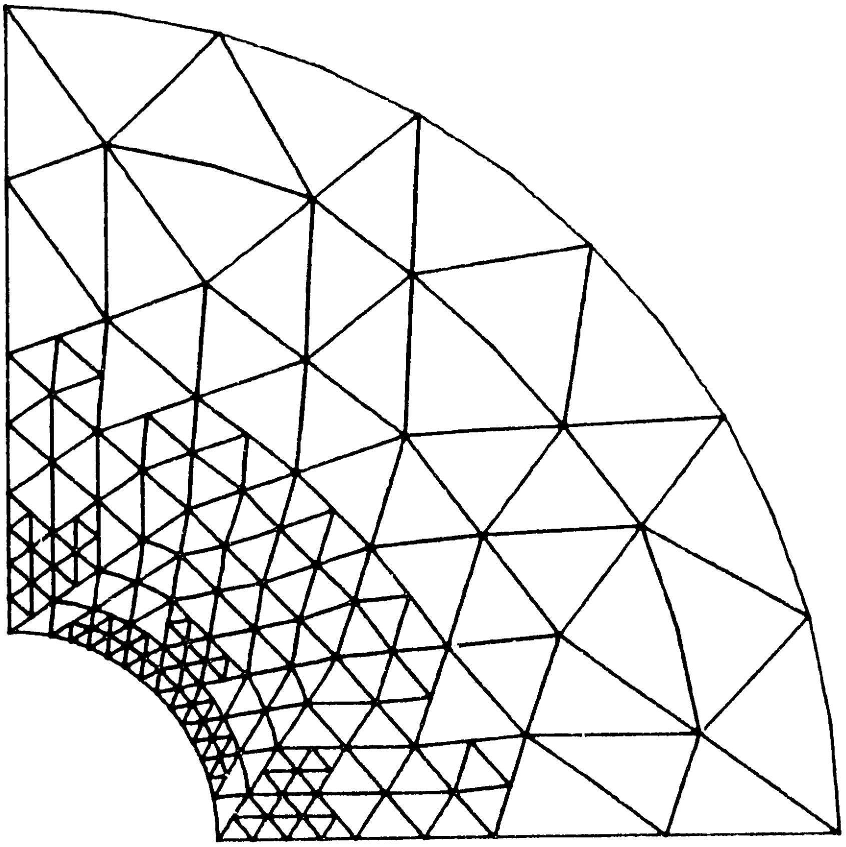


Figure 4-15: σ_{yy} distribution along the line A-A' for 17 element mesh



Total strain energy : 0.134696 x 10^5 N-m

ϵ_{total} = 0.09

$\epsilon_{tolerance}$ = 0.0002

(17 el → 68 el → 155 el → 227 el)

Figure 4-16: Final 239 element mesh after 3 steps of refinement

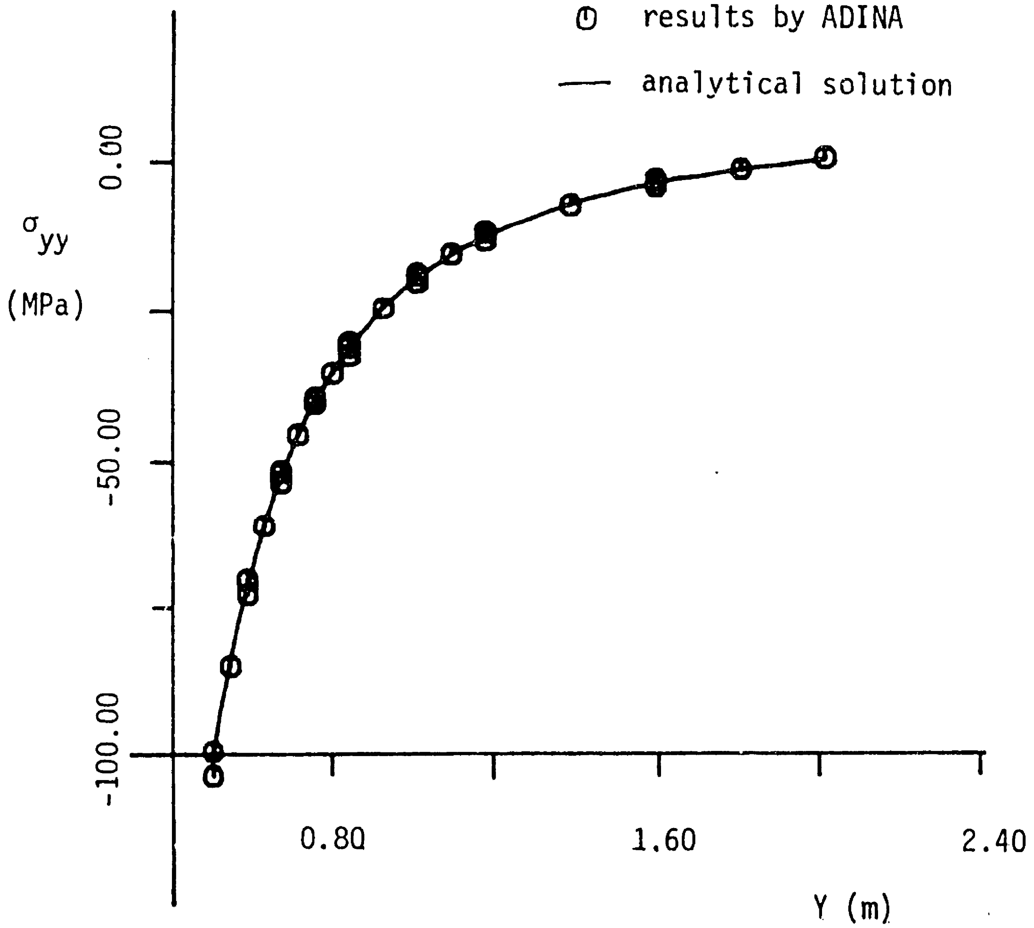


Figure 4-17: σ_{yy} distribution along the line A-A' for 239 elem:nt mesh

As a fine initial mesh, the 150 element mesh shown in Figure 4-18 is constructed. Using the same error tolerance, $\varepsilon = 0.002$, a 198 element mesh is obtained in 1 step (150 elements → 198 elements) as shown in Figure 4.19. The σ_{yy} distribution along the line A-A' is shown in Figure 4.20, which also compares favorably with the analytical solution. As expected in this example also, starting with a fine initial mesh mesh gives a more efficient mesh with less iterations.

(Example 4-3) Analysis of a Plate With Multiple Holes in Tension (Plane Stress)

As a plane stress problem, we consider the analysis of a plate with two holes in tension as shown in Figure 2.15. The goal of this analysis is to determine the stress distribution on the "lines of stress output", A-A' and B-B'. Since the analytical solution for this problem is not available, the stress discontinuities between adjacent elements is considered to determine the stress error involved in the solution.

The analysis domain is subdivided into 3 subdomains, see Figure 2-15, and a coarse initial mesh with 198 elements is constructed automatically as shown in Figure 4-21. The σ_{yy} distributions along the lines of interest, A-A' and B-B' are shown in Figure 4.22. The results in Figure 4.22 show that large discontinuities occur at the region of stress concentration, which means that the results are not satisfactory.

In order to obtain better results, the adaptive refinement is performed with an error tolerance, $\varepsilon_{tolerance} = 0.02$, and a 381 element model is obtained in 3 steps (198 elements → 303 elements → 375 elements → 381 elements) as shown in Figure 4-23.

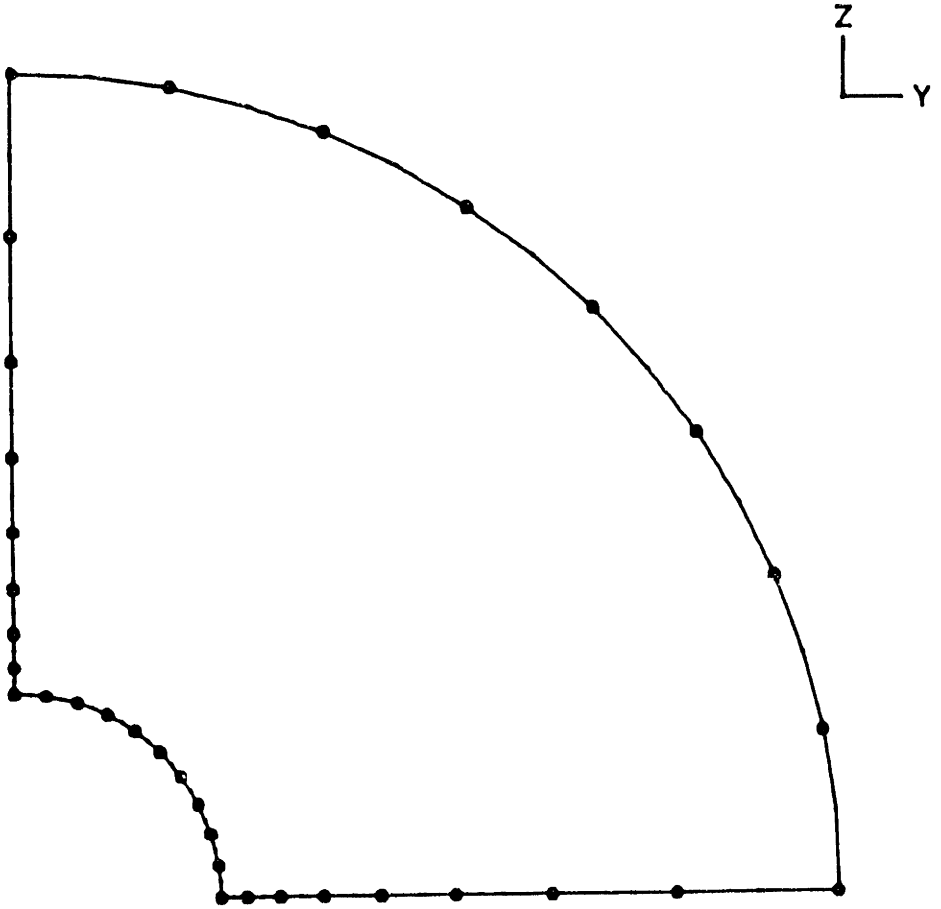
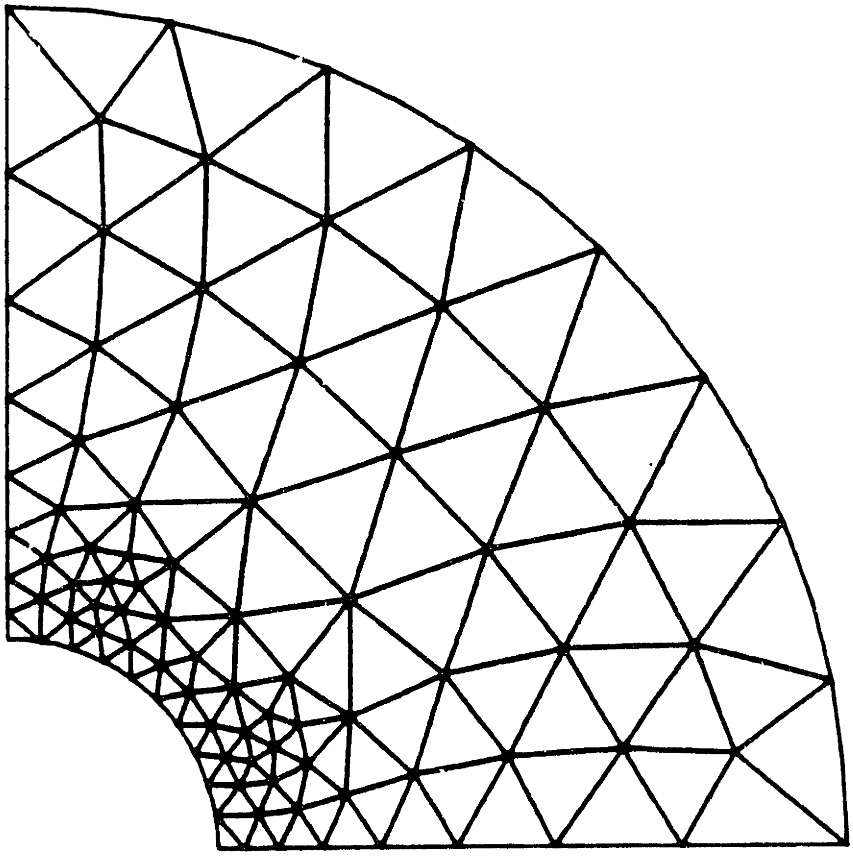


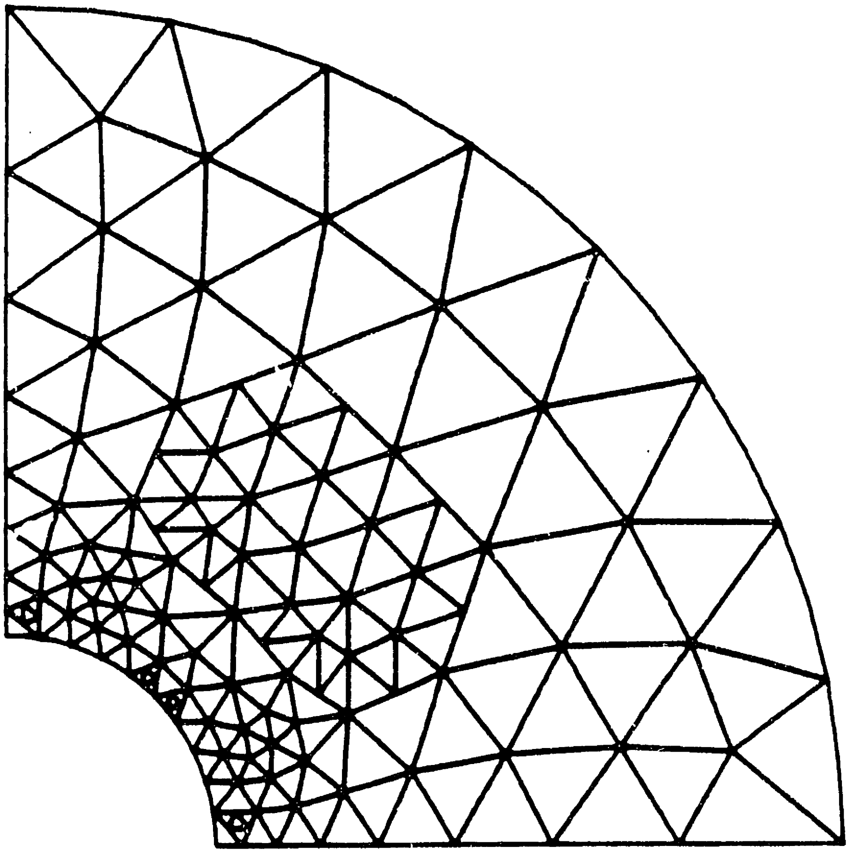
Figure 4-18 a): Key nodes distribution for a fine mesh generation



Total strain energy : 0.134793×10^5 N-m

$\epsilon_{total} = 0.16$

Figure 4-18 b): Fine starting mesh with 150 elements



Total strain energy : 0.134806 x 10^5 N-m

$\varepsilon_{total} = 0.10$

$\varepsilon_{tolerance} = 0.0002$

(150 el --> 198 el)

Figure 4-19 : Final 198 element mesh after one step of refinement

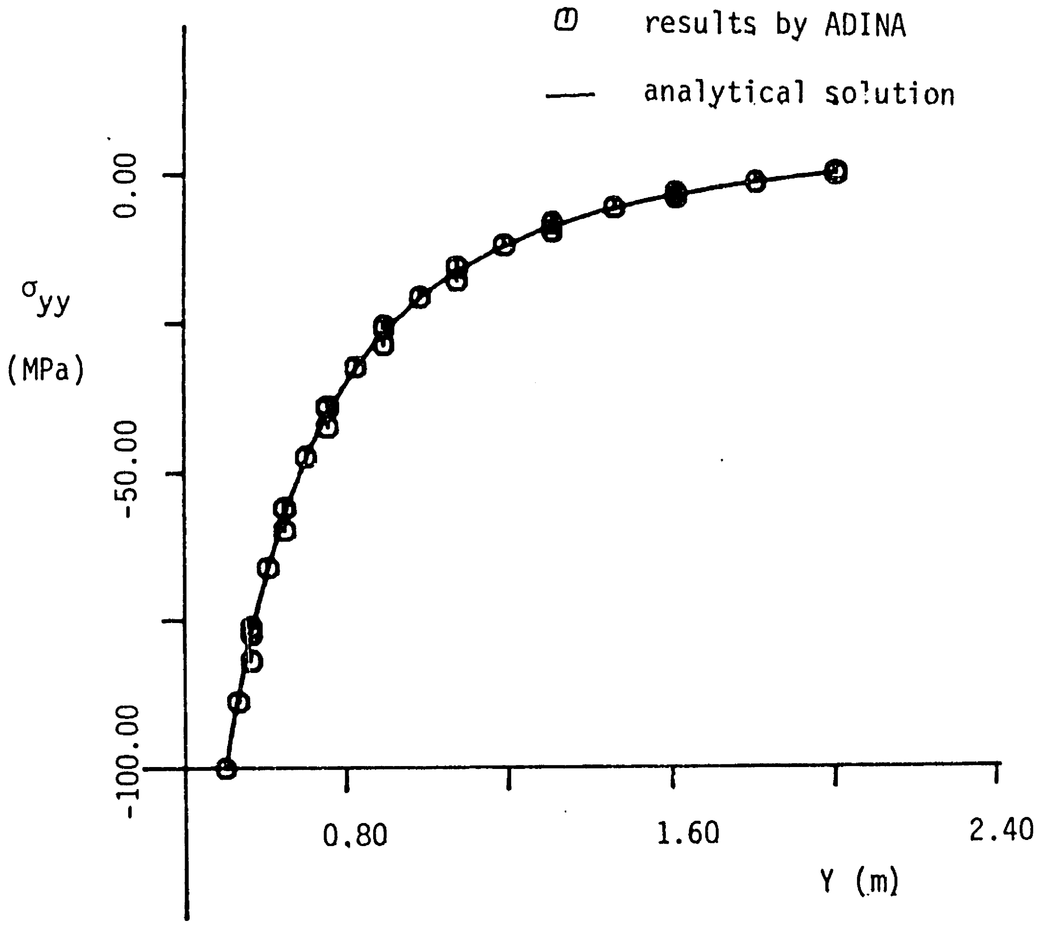


Figure 4-20: σ_{yy} distribution along the line A-A' for 198 element mesh

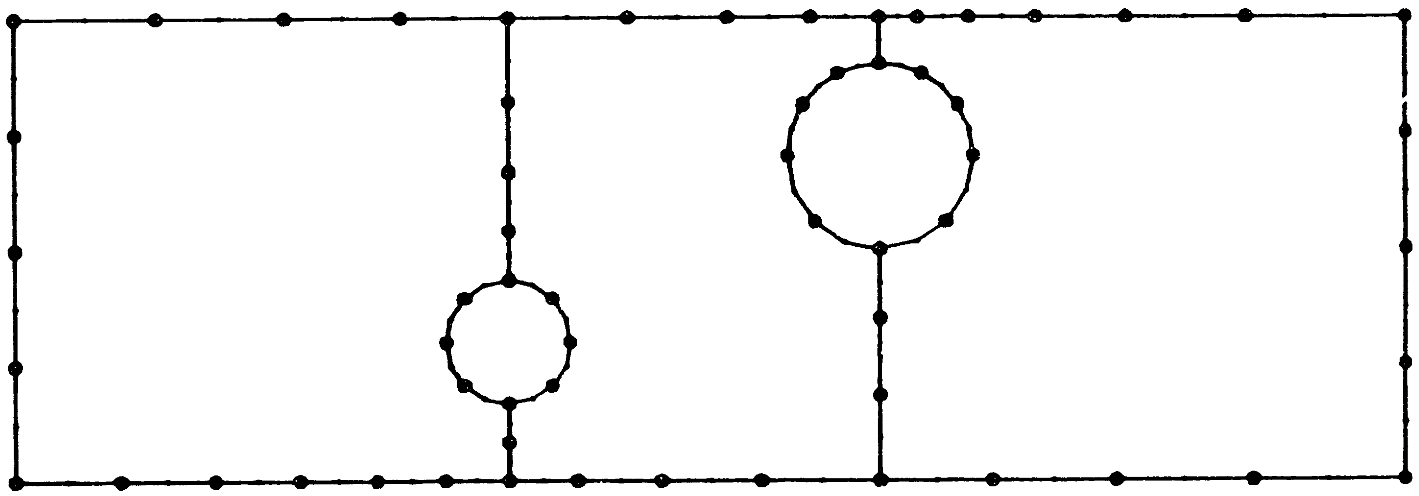
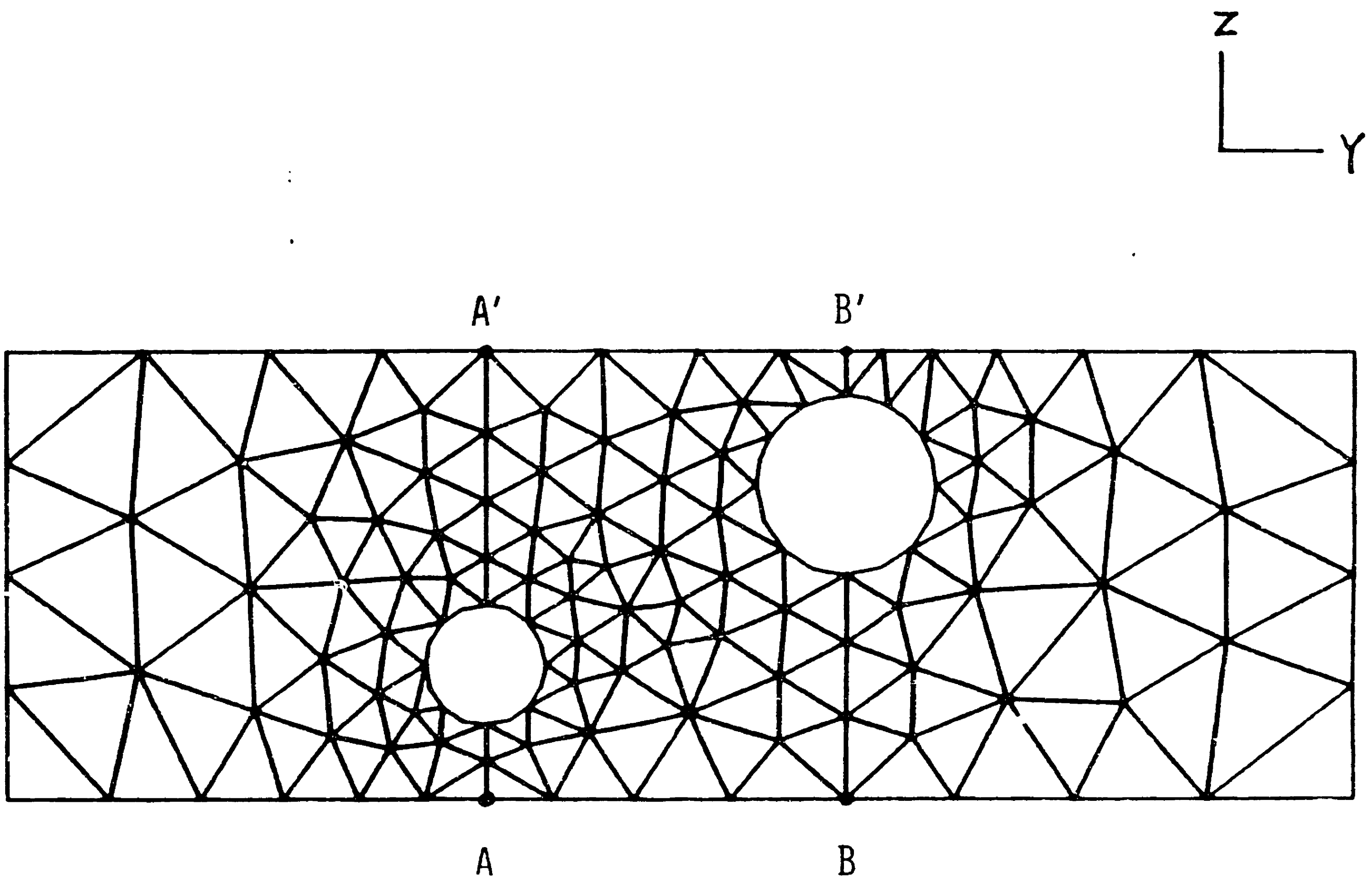


Figure 4-21 a): Key nodes distribution for a coarse mesh generation



Total strain energy : 0.814892 x 10^2 N-m

ε_{total} = 5.35

Figure 4-21 b): Coarse starting mesh with 198 elements

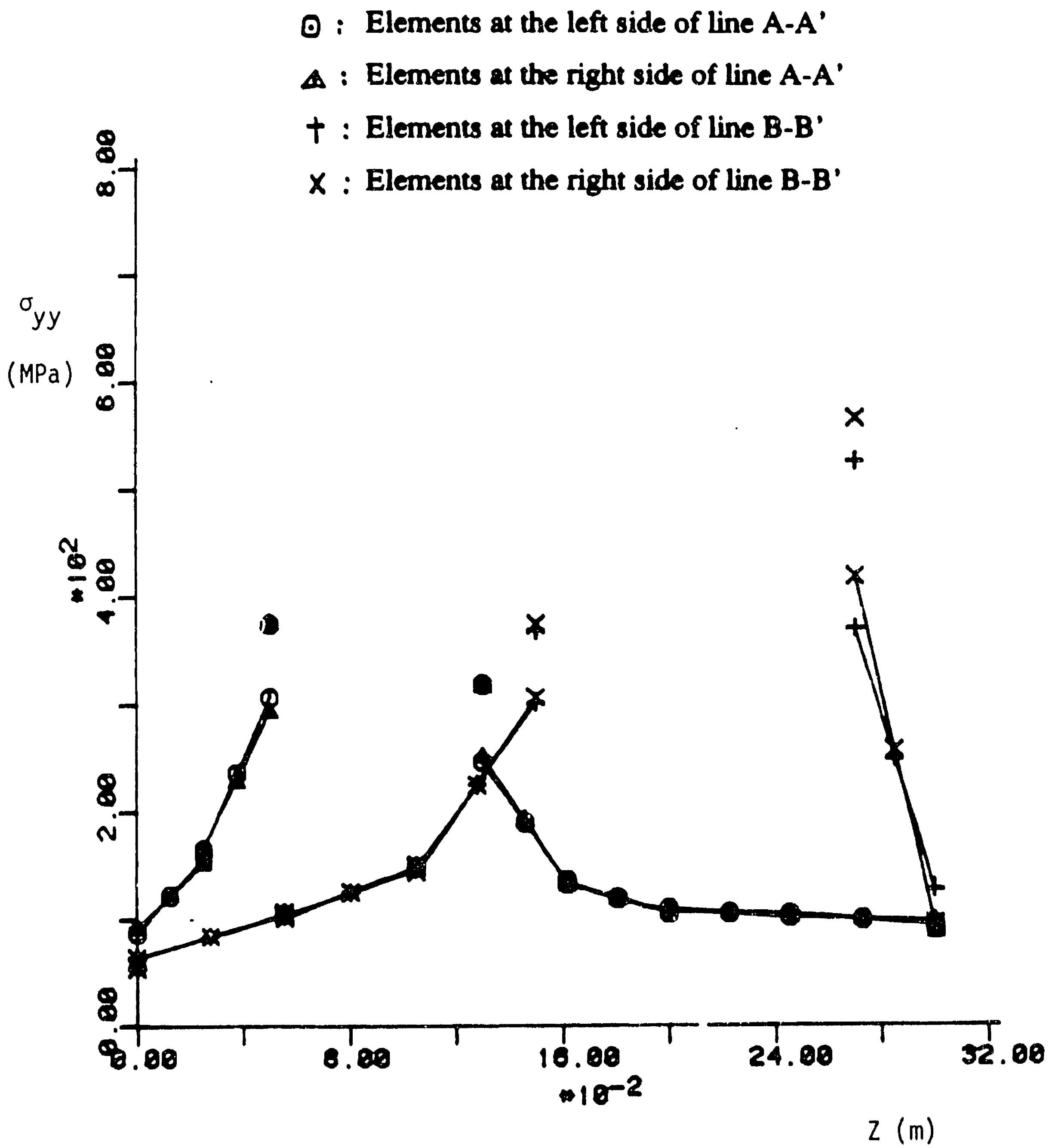


Figure 4-22: σ_{yy} distribution along the lines A-A' and B-B'

Total strain energy : 0.817691 x 10^2 N-m

$\varepsilon_{total} = 1.35$

$\varepsilon_{tolerance} = 0.02$

(198 el --> 303 el --> 375 el --> 381 el)

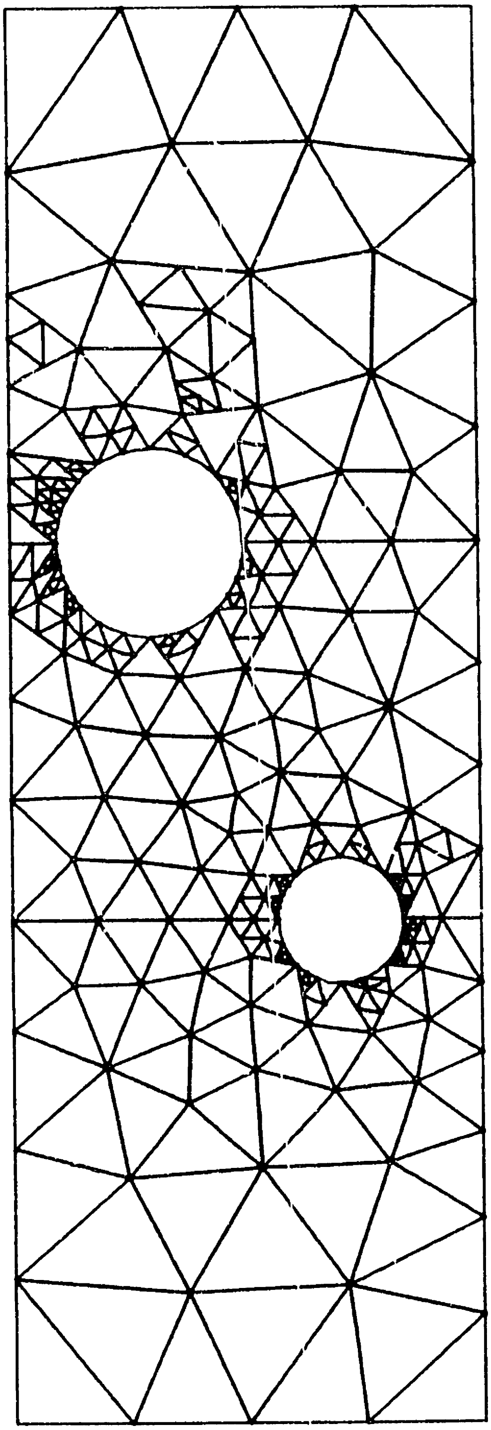


Figure 4-23: Final 381 element mesh after 3 steps of refinement

Here the third step of refinement may not be necessary, because only one element is refined in the third step. The stress output results for the 381 element mesh are shown in Figure 4-24, which show that the accuracy of solution has been improved significantly compared with the case of the 198 element mesh. If a more accurate solution is required, the adaptive refinement process should be performed further by reducing the tolerance of the error.

As a fine initial mesh, the 429 element mesh constructed in Figure 2-15 is employed for the analysis. By using the same error tolerance, $\varepsilon_{tolerance}$ = 0.02, a 495 element mesh is obtained in one step of the refinement (429 elements → 495 elements) as shown in Figure 4.25. The stress output results for this mesh are shown in Figure 4-26.

4.1.2 Examples Using Quadrilateral Elements

For the problems using 8-node quadrilateral elements, only the adaptive refinement process is implemented in our program. Hence the complete initial mesh should be input by the user, and then the adaptive refinement process is performed automatically by the program.

(Example 4-4) Analysis of an Axisymmetric Pressure Vessel

In this example, we consider again the analysis of an axisymmetric pressure vessel in Figure 2-21. The starting mesh with 69 eight-node quadrilateral elements shown in Figure 2-21 is employed for the adaptive refinement process. By using an error tolerance, $\varepsilon_{tolerance}$ = 0.03, the final mesh with 165 elements is obtained in 3 steps (69 element → 117element → 156 element → 165 element) as shown in Figure 4-27. This mesh compares well with the one by Sussman and

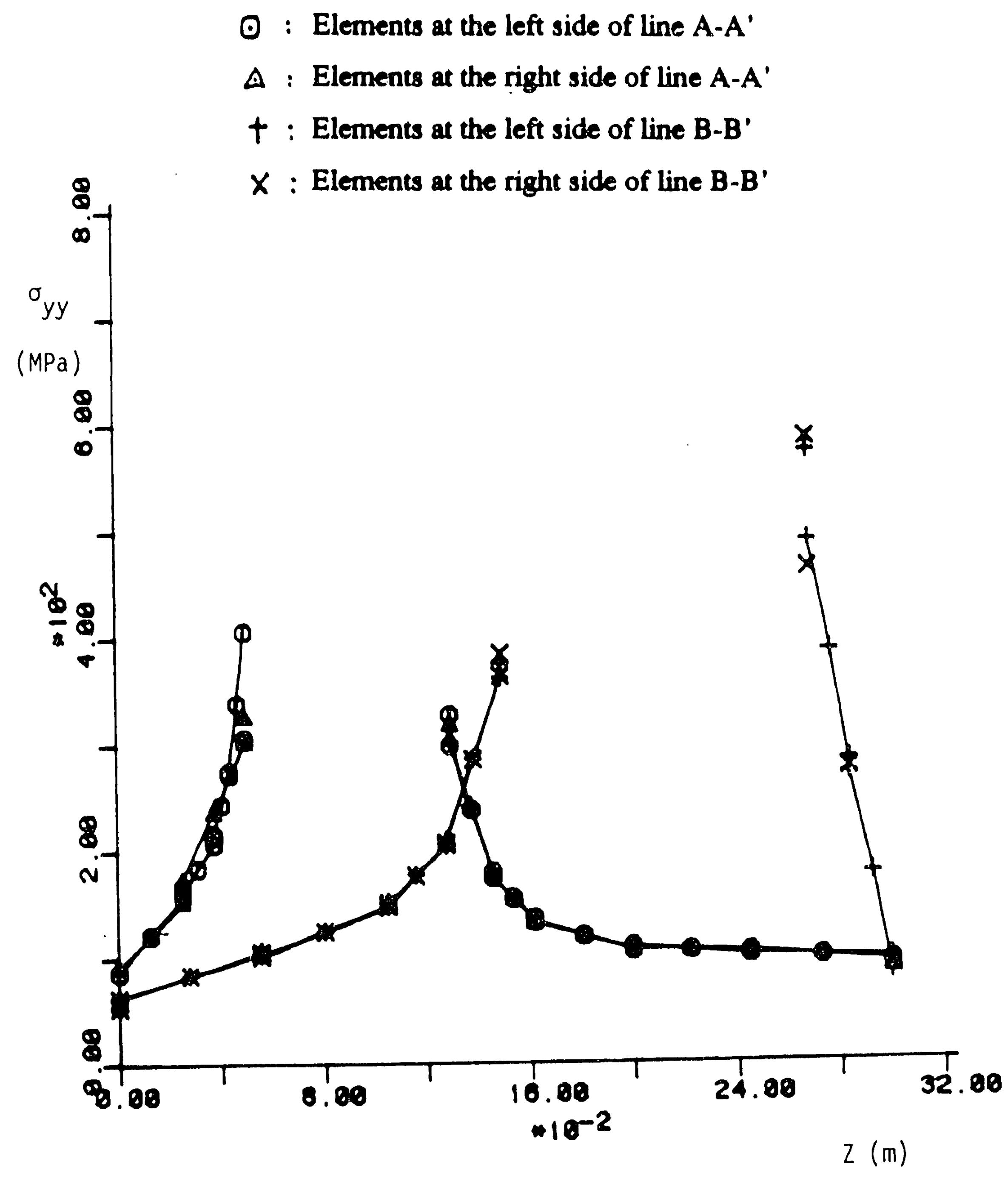
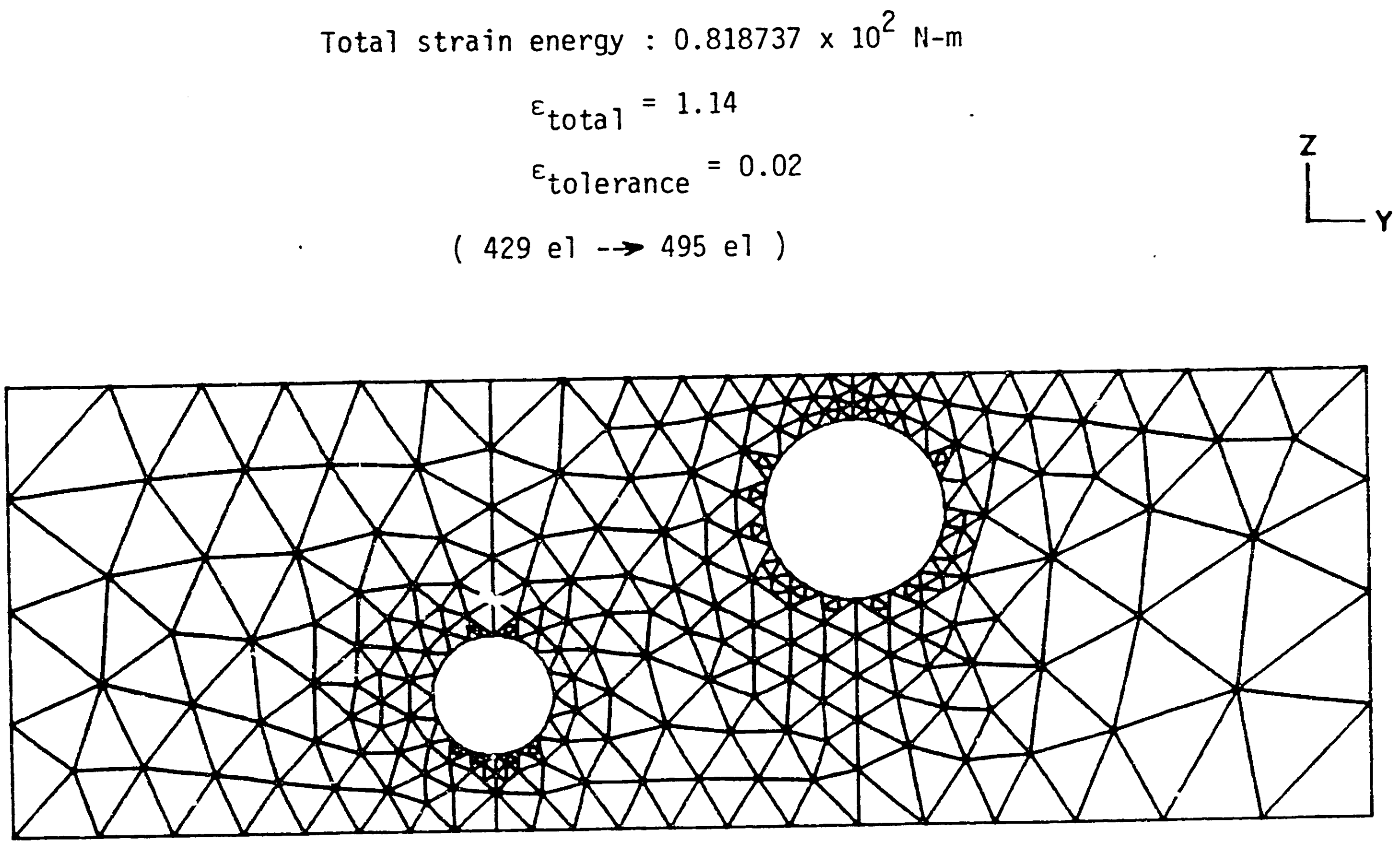


Figure 4-24: σ_{yy} distribution along the lines A-A' and B-B'



Total strain energy : 0.818737 x 10^2 N-m

$\varepsilon_{total} = 1.14$

$\varepsilon_{tolerance} = 0.02$

(429 el --> 495 el)

Figure 4-25: Final 495 element mesh after one step of refinement

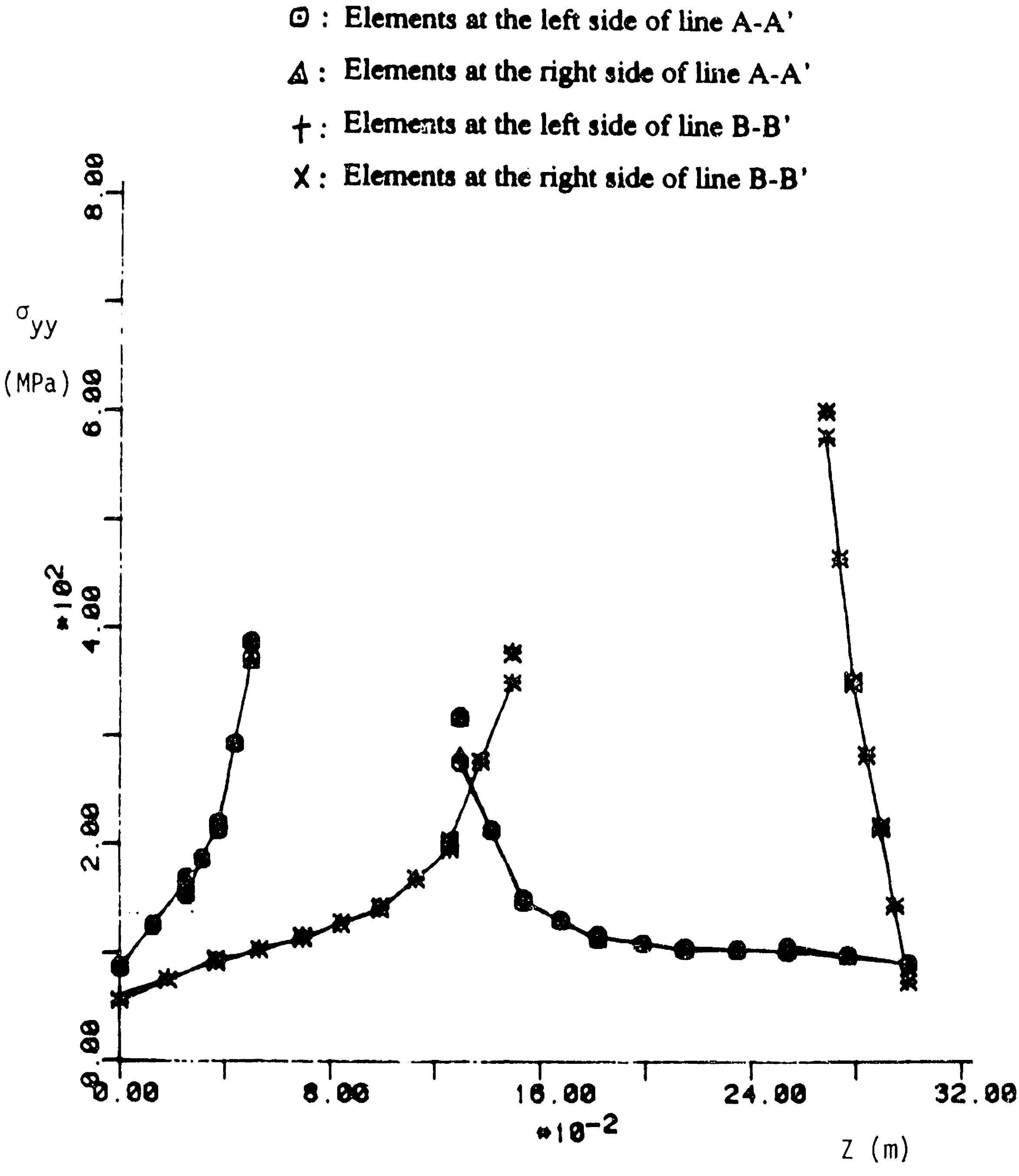
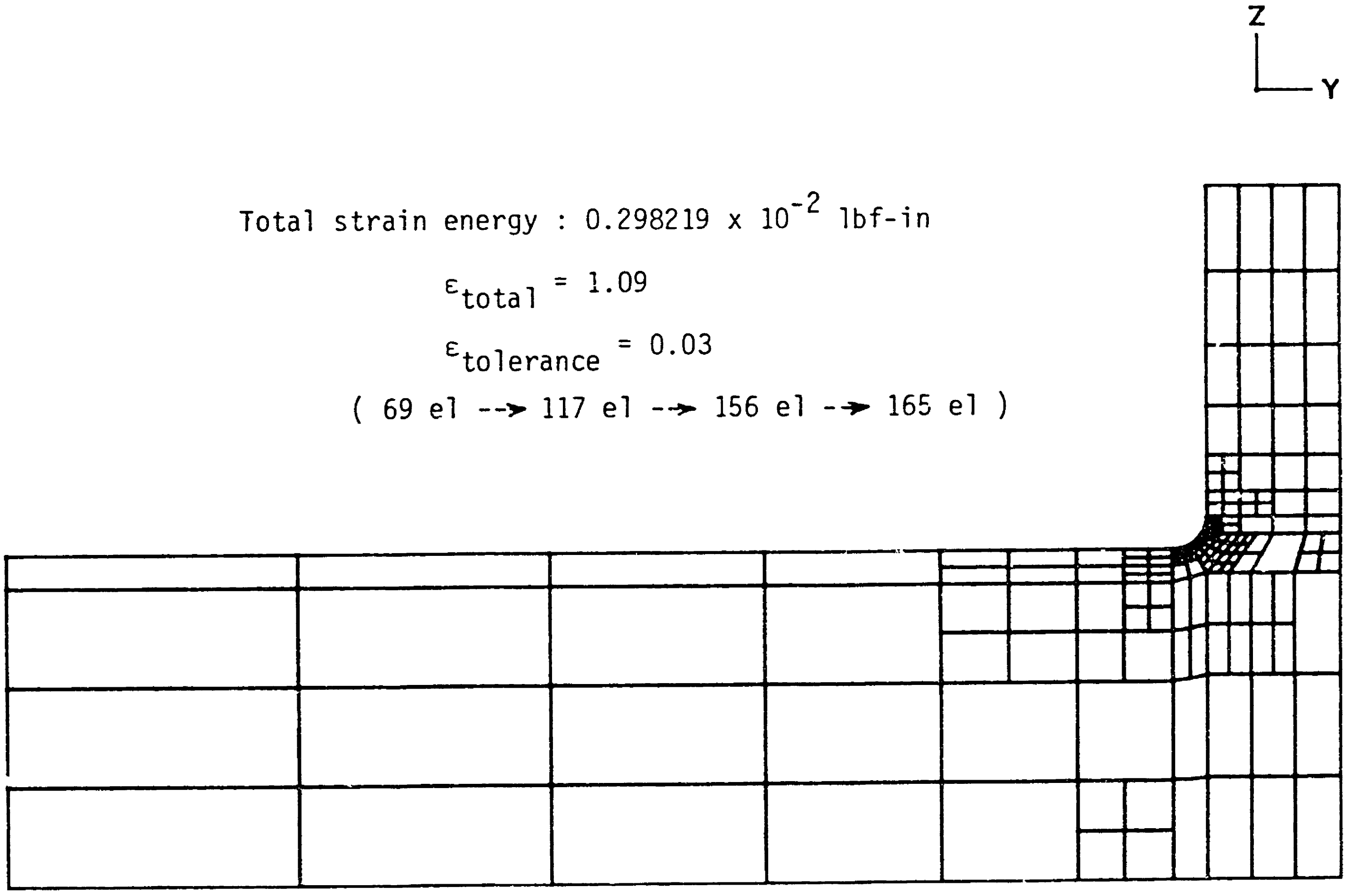


Figure 4-26: σ_{yy} distribution along the lines A-A' and B-B'



Total strain energy : 0.298219 x 10^{-2} lbf-in

$\varepsilon_{total} = 1.09$

$\varepsilon_{tolerance} = 0.03$

(69 el --> 117 el --> 156 el --> 165 el)

Figure 4-27 a): Final 165 element mesh after 3 steps of refinement

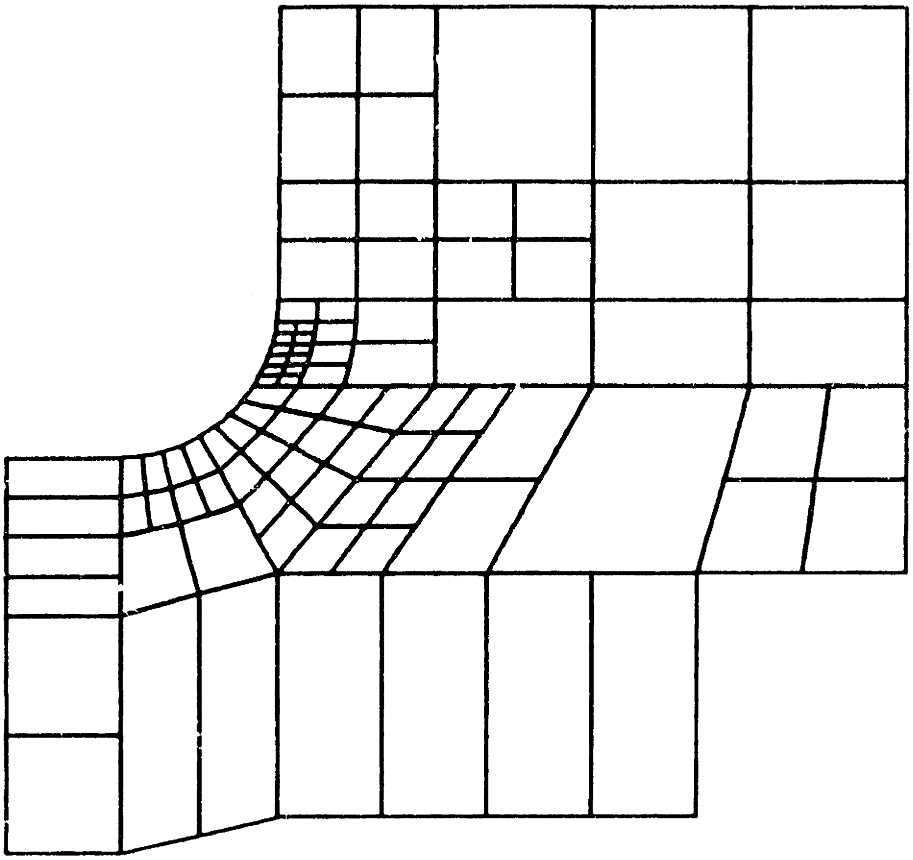


Figure 4-27 b): Detail of 165 element mesh

Bathe in Figure 4-1. The stress output results on the line of interest shown in Figure 4-28 show satisfactory results. Therefore we see that the adaptive refinement process using 8-node quadrilateral element is also very useful in constructing a near-optimal mesh.

(Example 4-5) Analysis of a Tensile Specimen With an Edge Crack

This example is to verify the performance of our adaptive refinement process using 8-node quadrilateral elements in linear elastic fracture mechanics analysis. We consider a tensile specimen with an edge crack, as shown in Figure 2-20, which is one of the verification examples in reference [63]. The goal of this analysis is to determine the stress intensity factor, K_I for the given geometry and load conditions.

The analytical solution for K_I is as follows:

$$K_I = \sqrt{\frac{E}{(1 - \nu^2)} \cdot G} \qquad (4-1)$$

and

$$G = \frac{-d\Pi}{dA}$$

where G = *energy release rate*

E = *Young's modulus*

ν = *Poisson's ratio*

dA = *change in crack area*

Π = *total potential energy*

The analytical value given in reference [64] is $K_I = 531.7$.

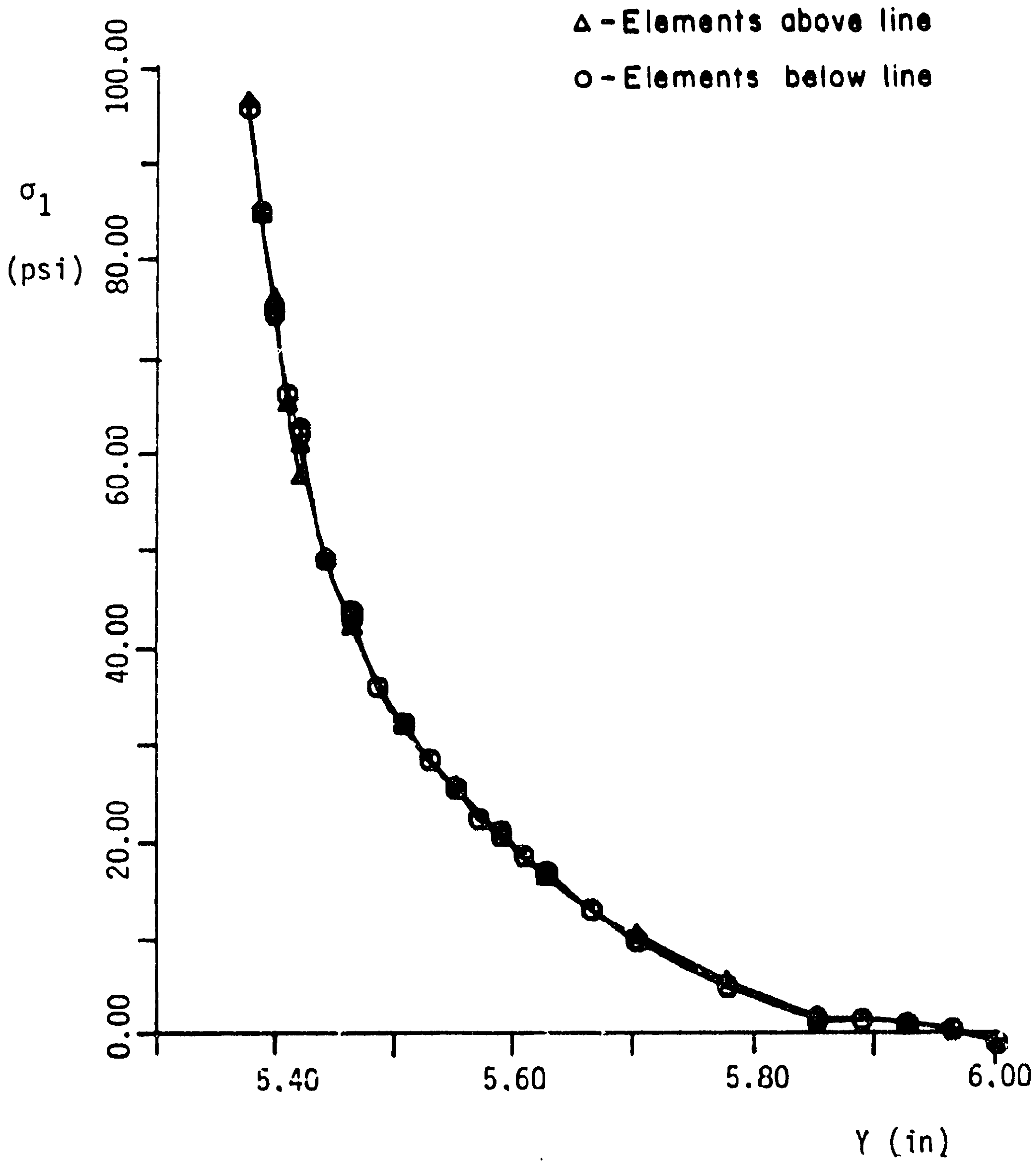


Figure 4-28 a): Maxımum in-plane principal stress for 165 element mesh

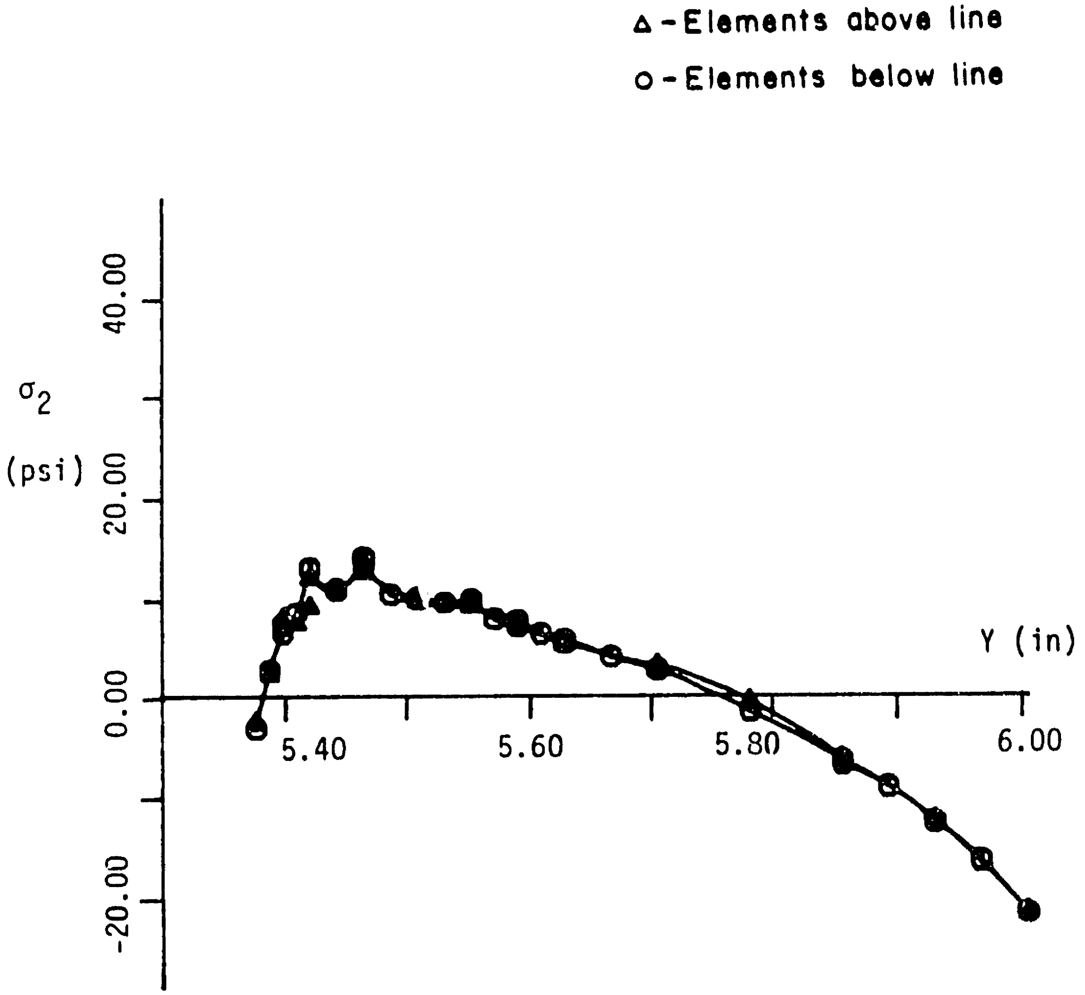


Figure 4-28 b): Minimum in-plane principal stress for 165 element mesh

The strain energy release rate at the crcak tip node is obtained as [4]

$$G = - \frac{\partial \Pi}{\partial x_i^c} \frac{a_i^c}{t} \qquad (4\text{–}2)$$

where x_i^c is the coordinate of the crack tip nodal point, a_i^c is the component of the unit vector in the direction of crack propagation, and t is the specimen thickness. Ths ADINA numerical solution gives the value of $\partial \pi / \partial x_i^c$ and thus we can calculate G and K_I.

Initially, a 12 element model shown in Figure 2-20 is used for the analysis and the error indicators computed are shown in Figure 4-29. The results in Figure 4-29 show that the values of the error indicator around the crack tip node are large. The stress intensity factor, K_I for 12 element model is obtained as 455.4. In order to obtain a better solution, the adaptive refinement process is performed using an error tolerance, $\varepsilon_{tolerance} = 0.02$. After five steps of refinement process (12 element → 24 element → 42 element → 60element → 78 element → 96 element), the resulting mesh with 96 elements is constructed as shown in Figure 4-30. In this example, the adaptive refinement is stopped after five steps, although in some elements the error indicators are still larger than the given tolerance.

The results of the error analysis for the first refined mesh, are shown in Figure 4-31, and the results for the meshes used in the refinement procses are summarized in Table 4-1.

One interesting point to note is that the error indicators around the crack tip do not decrease , instead they slightly increase during the refinement process, while in the other region they decrease rapidly.

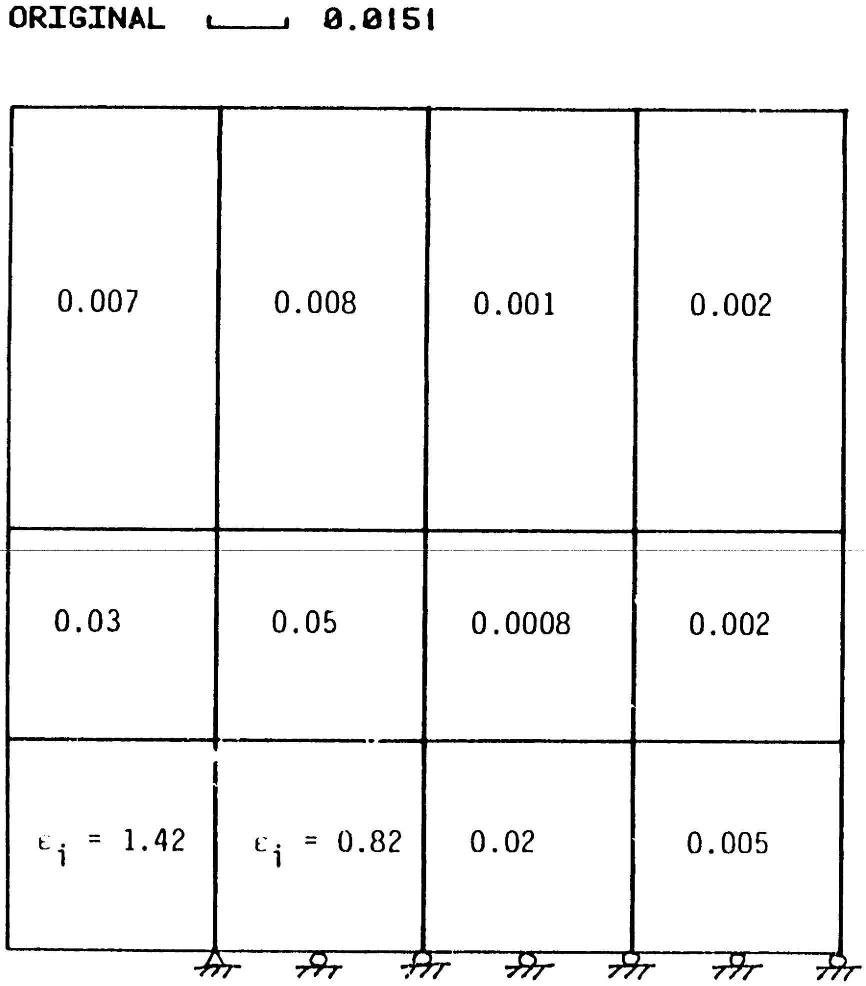
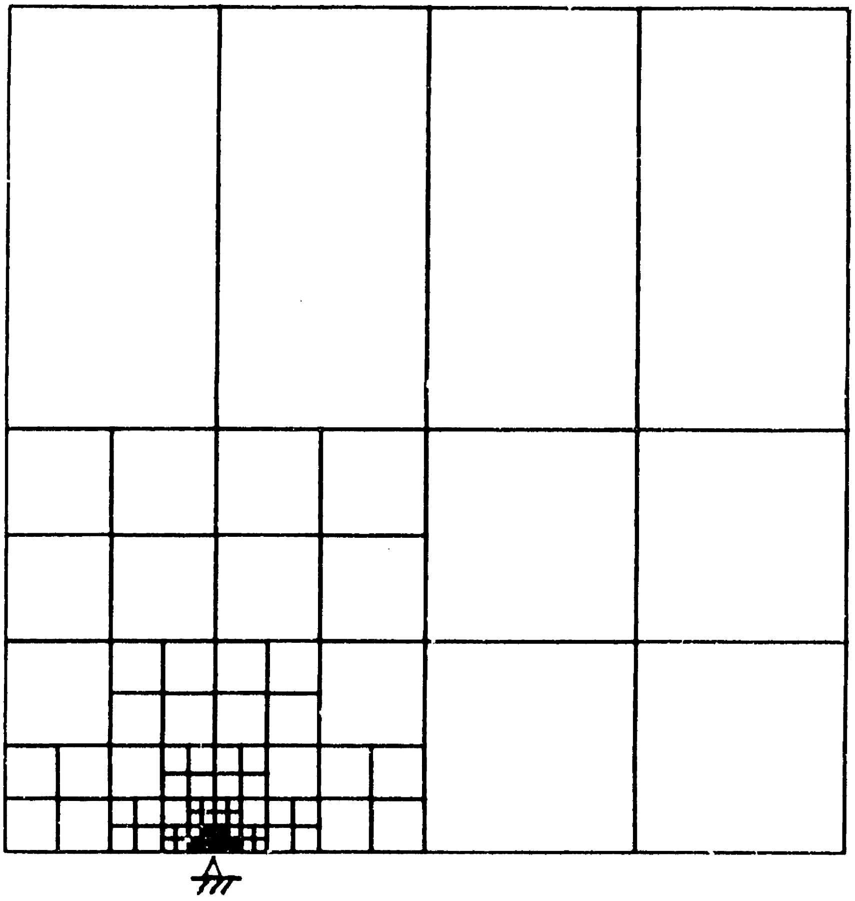


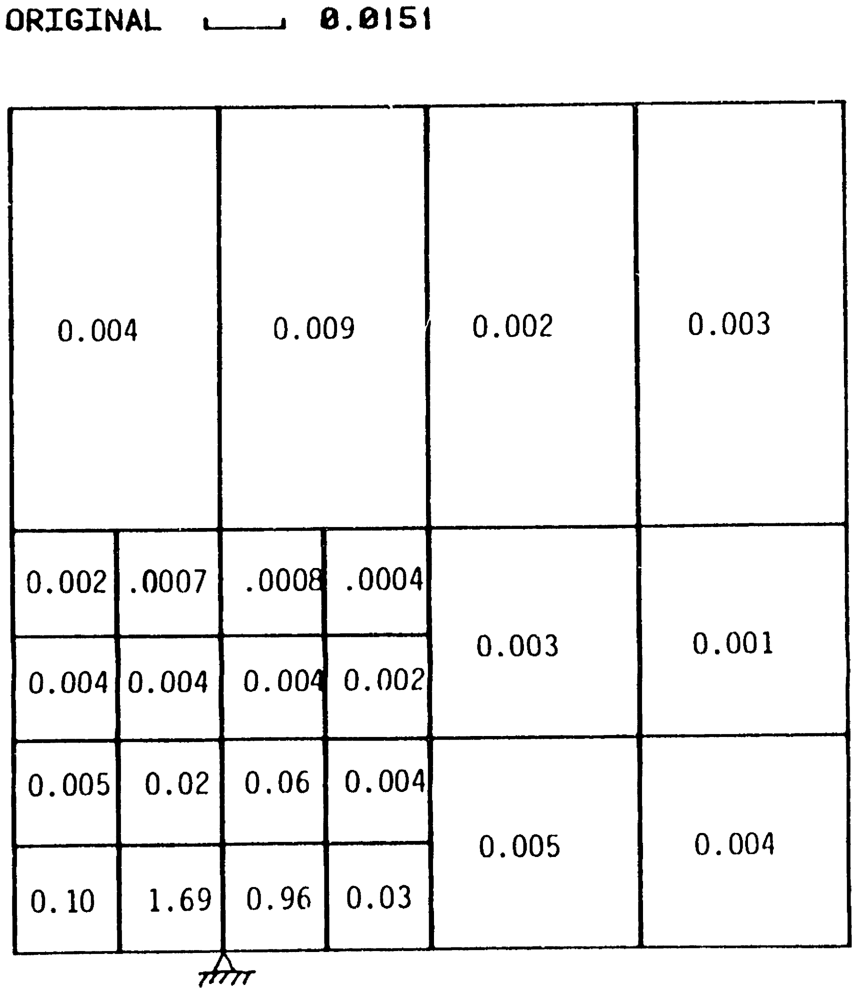
Figure 4-29: Results of the error analysis for 12 element mesh



$\varepsilon_{\text{tolerance}} = 0.02$

(12 el --> 24 el --> 42 el --> 60 el --> 78 el --> 96 el)

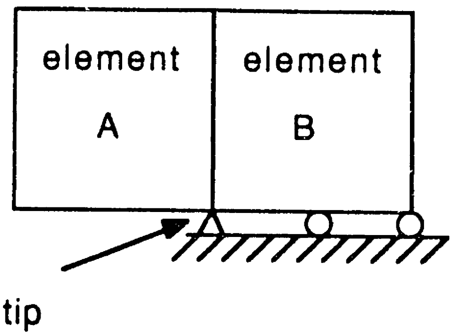
Figure 4-30: 96 element mesh: 96 element mesh after 5 steps of refinement



(12 el --> 24 el)

Figure 4-31: Error analysis for 24 element mesh after the first refinement

Table 4-1. K_I values for various meshes



number of elements	K_I	ε_A	ε_B
12	455.4	1 42	0.81
24	482.8	1.69	0.96
42	497.2	1.81	1.04
60	504.2	1.86	1.07
78	507.7	1.88	1.09
96	509.4	1.89	1.10
Exact	531.7		

The stress intensity factor obtained with the 96 element mesh is 509.4. Here the convergence rate of K_I to the analytical solution seems to be slow, because the singular elements [58] are not employed.

4.2 Three-Dimensional Application

In this section, two examples are considered in order to test the validity of our three-dimensional mesh generation and error analysis scheme. One example is similar to a conventional two-dimensional problem, a thin plate with a hole in tension, and the other is a general three-dimensional problem, a block with a cylinder.

For the three-dimensional analysis using ADINA, the 20-node degenerated solid element is used, which is equivalent to a 10-node tetrahedral element. 3×3 Gauss integration is employed for the analysis and the stresses are computed directly at the nodal points.

(Example 4-6) Analysis of a Thin Plate with a Cylindrical Hole

In this example, a thin plate with a hole in tension as shown in Figure 4-32 is analyzed by using the three-dimensional tetrahedral elements. This problem is usually modeled as a two-dimensional plane stress problem.

The goal of this analysis is to determine the stress concentration factor on the line A-A'. The stress concentration factor for this problem in plane stress condition falls around 3.4 [65], although the exact analytical value is not available due to the finite width of the plate in the directions of x and y.

Three different meshes are constructed by considering one quarter of

$E = 207000$ MPa

$\nu = 0.3$

$R = 0.2$ m

$L = W = 2$m , $t = 0.1$ m

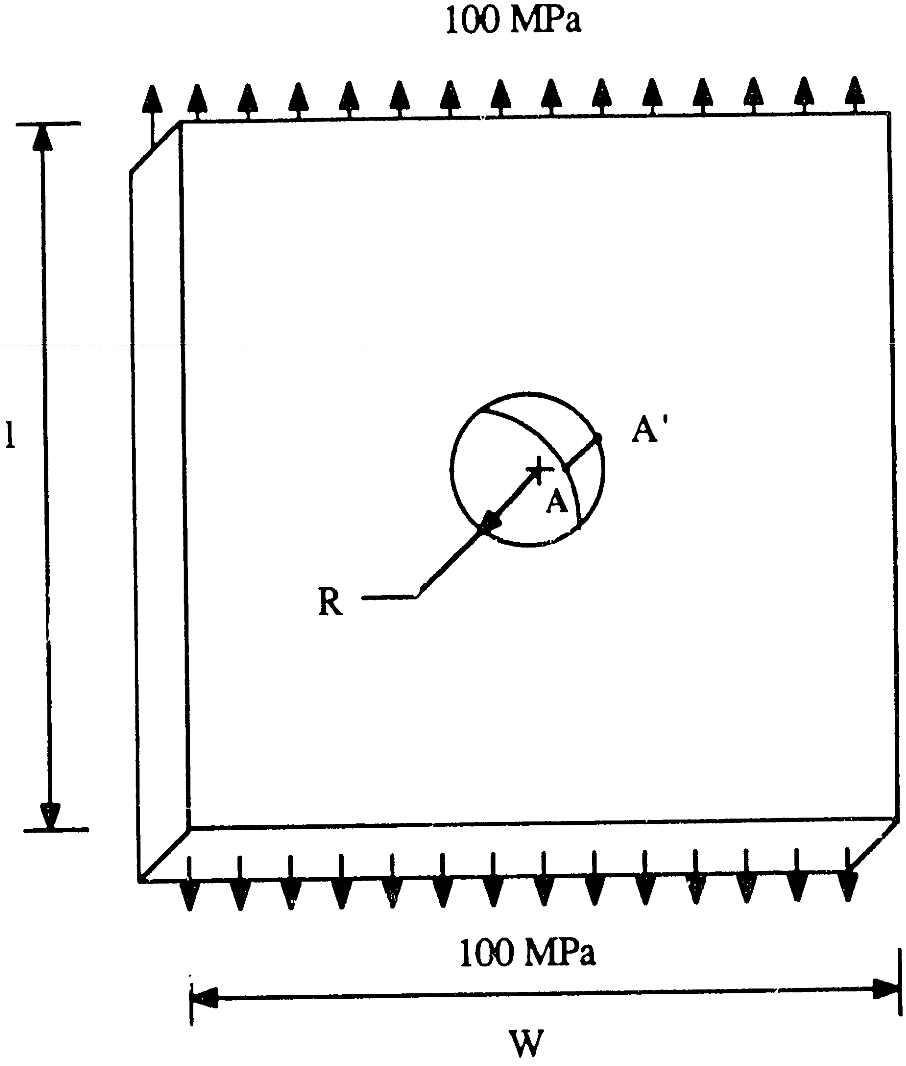


Figure 4-32: Thin plate with a hole in tension

the analysis domain due to symmetry. To begin, the meshes are constructed only with the straight-sided elements. The first mesh is a uniform coarse mesh with 39 elements as shown in Figure 4-33. The analysis results are shown in Figure 4-34. As shown in Figure 4-34, the maximum stresses are obtained in the element 8 as expected, where the stress concentration occurs and the average stress concentration factor on the line A-A' is obtained as 2.06. The largest error indicators are obtained in the elements - 6, 8, 18, which indicate where refinement is necessary.

The second mesh with 180 elements is a uniform finer mesh than the first one as shown in Figure 4-35. For this mesh, the maximum stresses are obtained in the element 10, and the average stress concentration factor along A-A' is obtained as 2.75. The largest error indicators are obtained in the element 10 ($\varepsilon_{10} = 0.16$), 38 ($\varepsilon_{38} = 0.076$), 37 ($\varepsilon_{37} = 0.035$), which show where the refinement is necessary.

The third mesh is a graded mesh constructed with 108 elements as shown in Figure 4-36. The maximum stresses are obtained in the element 2, and the average stress concentration factor along A-A' is obtained as 3.73, which is larger than the value in the two-dimensional problem. The largest error indicators are obtained in the element 29 ($\varepsilon_{29} = 0.066$), and 28 ($\varepsilon_{28} = 0.033$).

The results from the above three meshes are summarized in Table 4-2. As shown in Table 4-2, a graded mesh has the largest strain energy and the least total error indicator, although less elements (108 elements) are used than the mesh with 180 elements. This shows the efficiency of the mesh employed.

However, since only the straight-sided tetrahedral elements are employed in these analyses, the cylindrical surface of a hole is not accurately modeled, which will affect the accuracy of the solution.

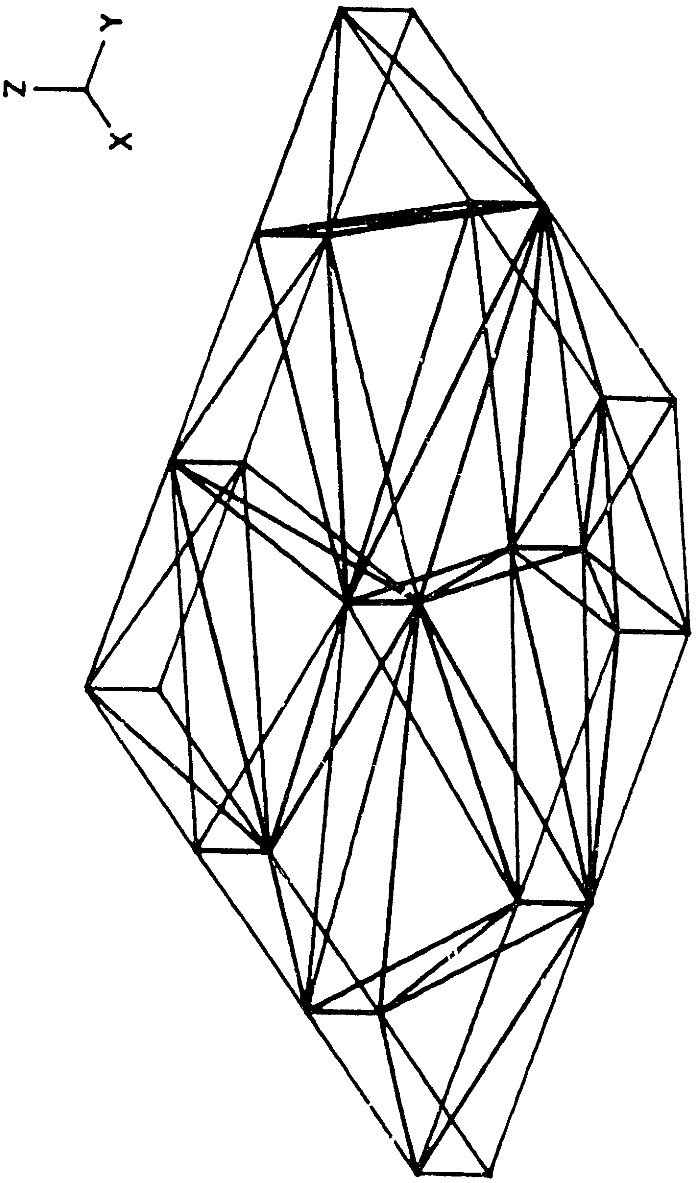
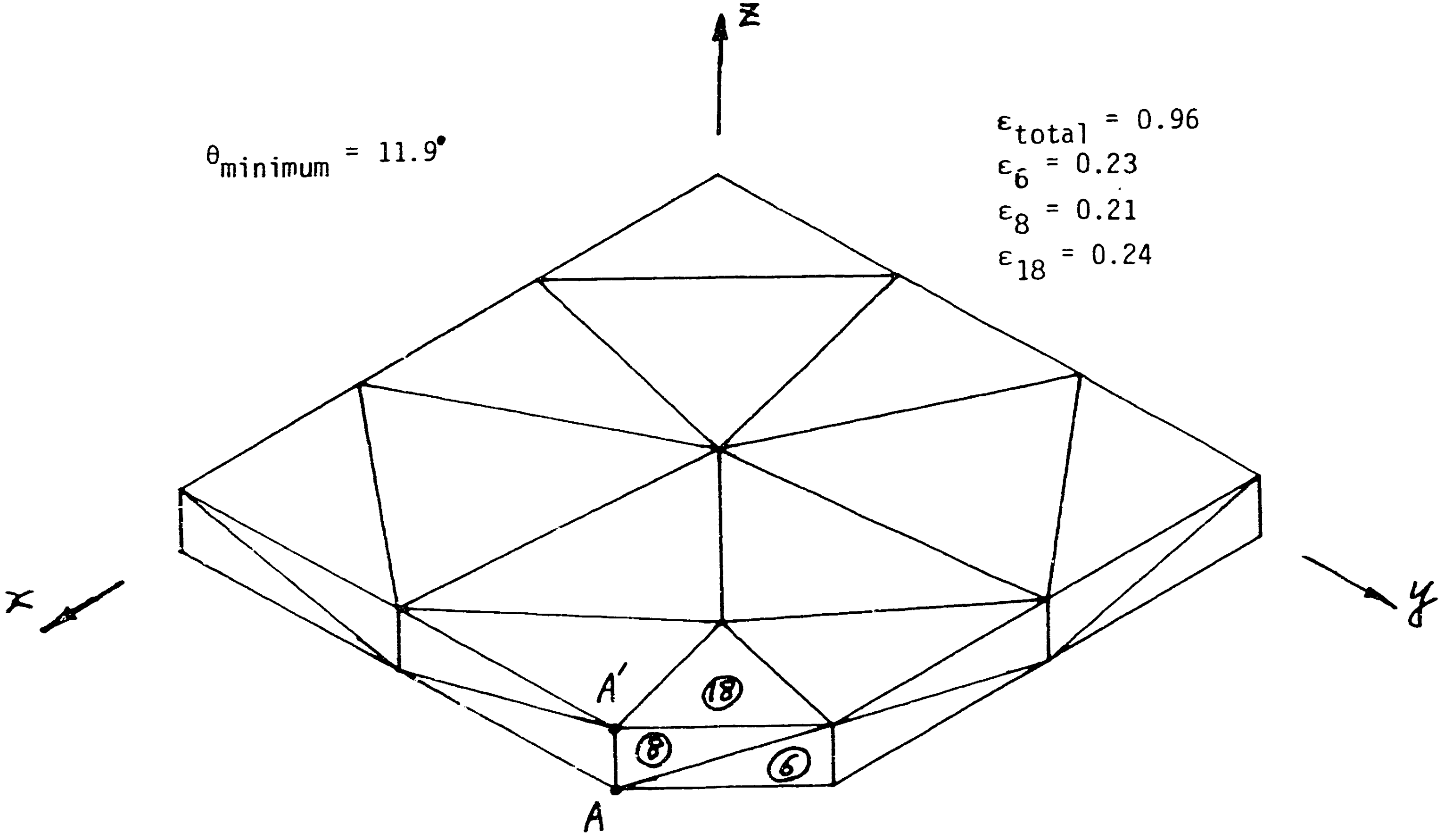


Figure 4-33: 39 element mesh with straight-sided elements



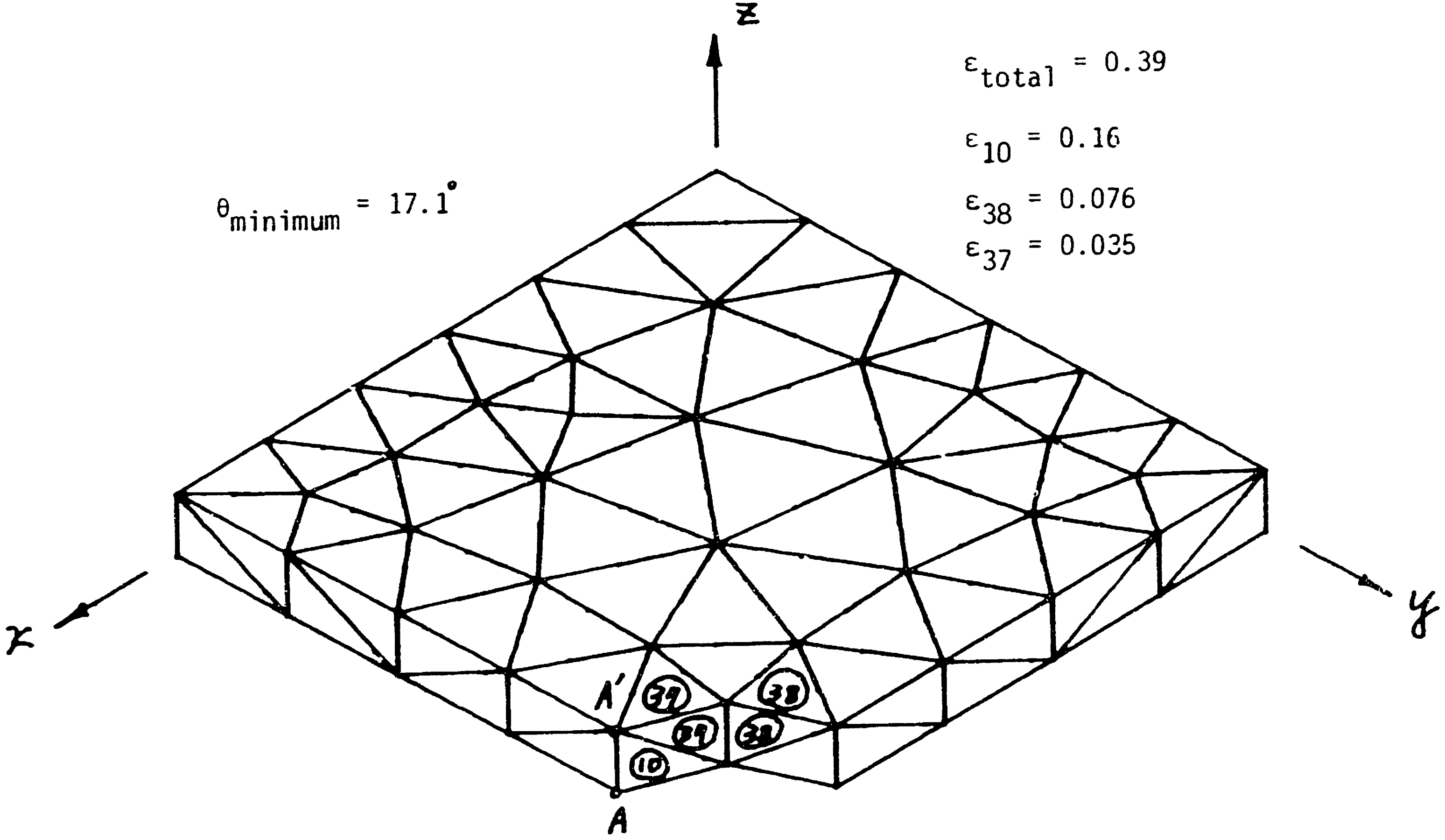
$\theta_{minimum} = 11.9°$

$\varepsilon_{total} = 0.96$
$\varepsilon_{6} = 0.23$
$\varepsilon_{8} = 0.21$
$\varepsilon_{18} = 0.24$

Total strain energy : 0.258 x 10^4 N-m

$\sigma_{xx}\big|_{A-A'} \cong 206$ MPa

Figure 4-34: Results of 39 element mesh with straight-sided elements



$\theta_{minimum} = 17.1^{\circ}$

$\varepsilon_{total} = 0.39$

$\varepsilon_{10} = 0.16$

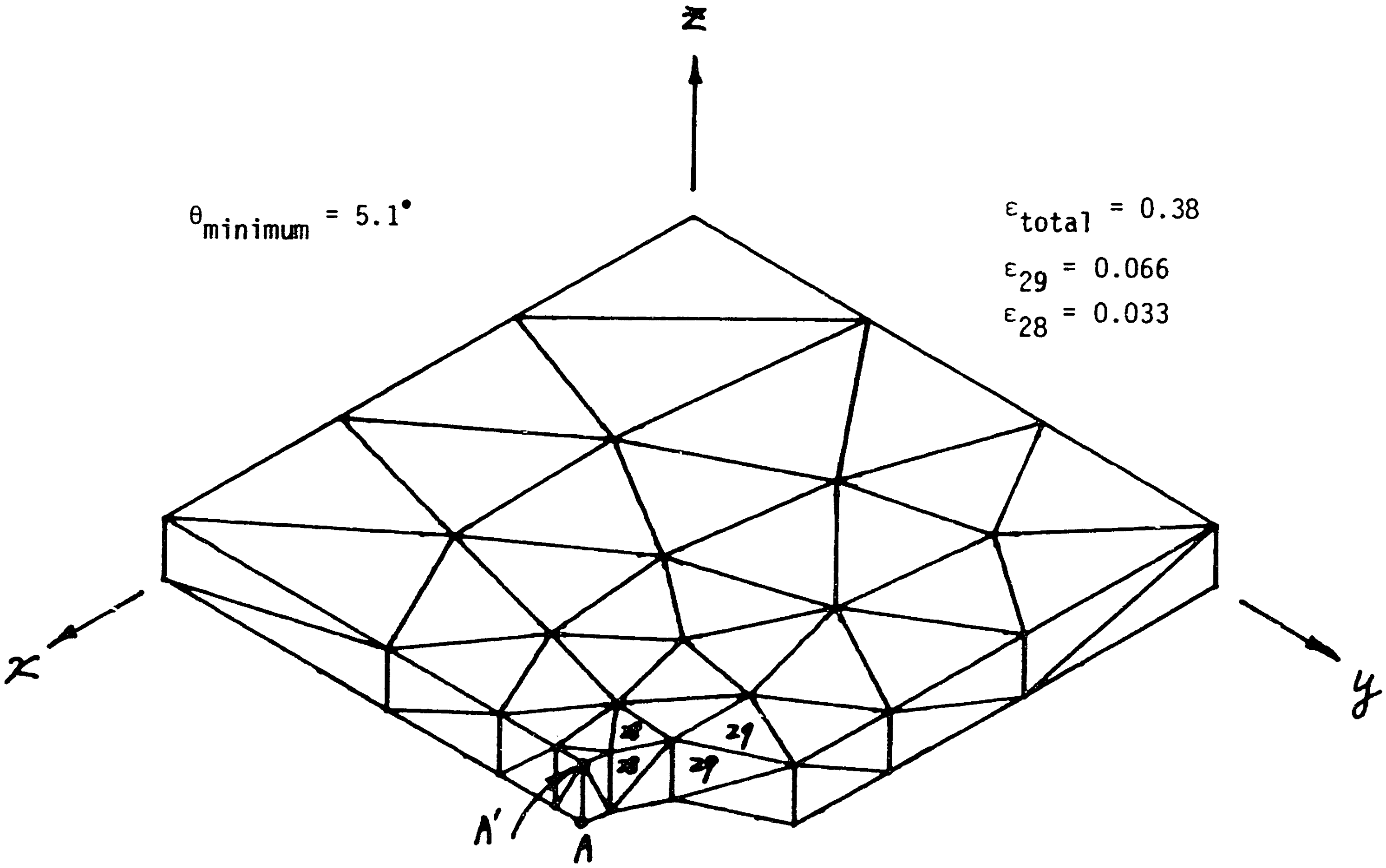
$\varepsilon_{38} = 0.076$

$\varepsilon_{37} = 0.035$

Total strain energy : 0.264×10^{4} N-m

$\sigma_{xx}\big|_{A-A'} \cong 275$ MPa

Figure 4-35: 180 element mesh with straight-sided elements


z
$\theta_{minimum} = 5.1°$
$\varepsilon_{total} = 0.38$
$\varepsilon_{29} = 0.066$
$\varepsilon_{28} = 0.033$
x
y
28
28
29
29
A'
A

Total strain energy : 0.266×10^4 N-m

$\sigma_{xx}\Big|_{A-A'} \cong 364$ MPa

Figure 4-36: 108 element mesh with straight-sided elements

Table 4-2. The stress concentration factors for the meshes with straight-sided elements.

	Strain Energy N-m	Average Stress Concentration Factor along A-A'	Total Volume m^3	Total Error Indicator ε_{total}
39 elements	0.258×10^4	2.06	0.0980	0.96
180 elements	0.264×10^4	2.75	0.0972	0.39
108 elements	0.266×10^4	3.64	0.0970	0.38
Reference Value (2-D plane stress condition)		3.4		

In order to model the cylindrical surface of a hole more accurately, curved-sided elements are again employed for the same meshes as above, in which only the mid-side nodes on the cylindrical surface are relocated to fall on the cylindrical surface. The corresponding meshes constructed are shown in Figure 4-37 to Figure 4-39 and the results of the analyses are summarized in Table 4-3. In Table 4-3, we see that the results are improved by using a fine mesh or a graded mesh. The average stress concentration factor obtained by the 108 element model is 3.43, which compares well with the reference value obtained in the two-dimensional analysis.

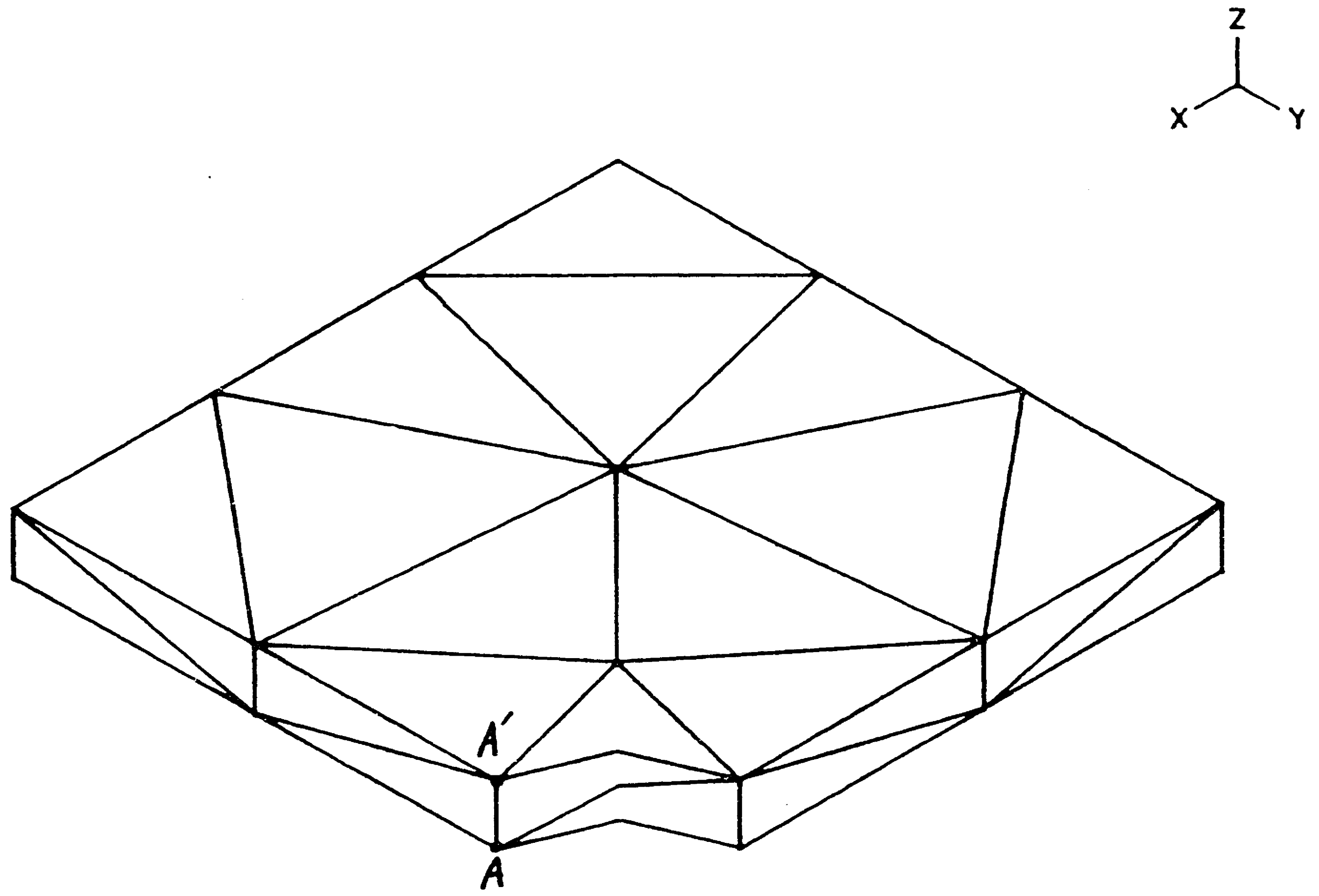
(Example 4-7) Analysis of a cylinder attached to a block

As a general three-dimensional problem, we consider a cylinder attached to a block under two loading conditions, i.e., bending and torsion as shown in Figure 4-40. For the analysis of this problem, three different meshes are constructed, ranging from a coarse one to a fine one. In order to construct each mesh with well-conditioned elements, the analysis object is subdivided into convex subobjects especially at the region of high local mesh density.

For a coarse mesh construction, the analysis object is subdivided into three convex subobjects as shown in Figure 4-41, while for a fine mesh construction it is subdivided into six subobjects as shown in Figure 4-42.

As a result of the volume triangulation for every subobject, three different meshes are constructed with 514 elements, 1003 elements, and 1568 elements as shown in Figure 4-43, Figure 4-44, and Figure 4-45 respectively.

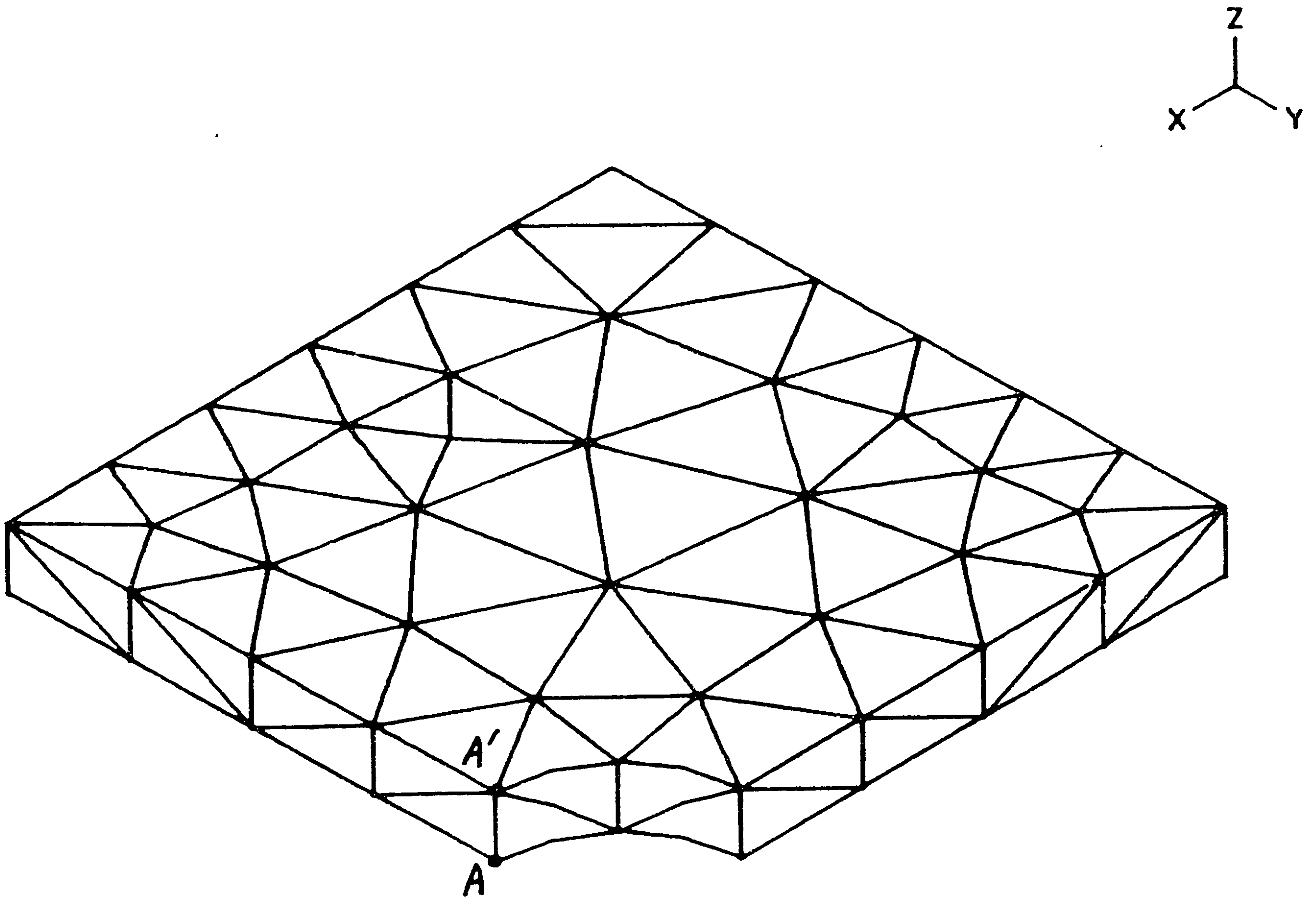
In Figure 4-43, object-1 and object-2 are originally same with each



Total strain energy : 0.264 x 10^4 N-m

$\sigma_{xx}\big|_{A-A'} \cong 227$ MPa

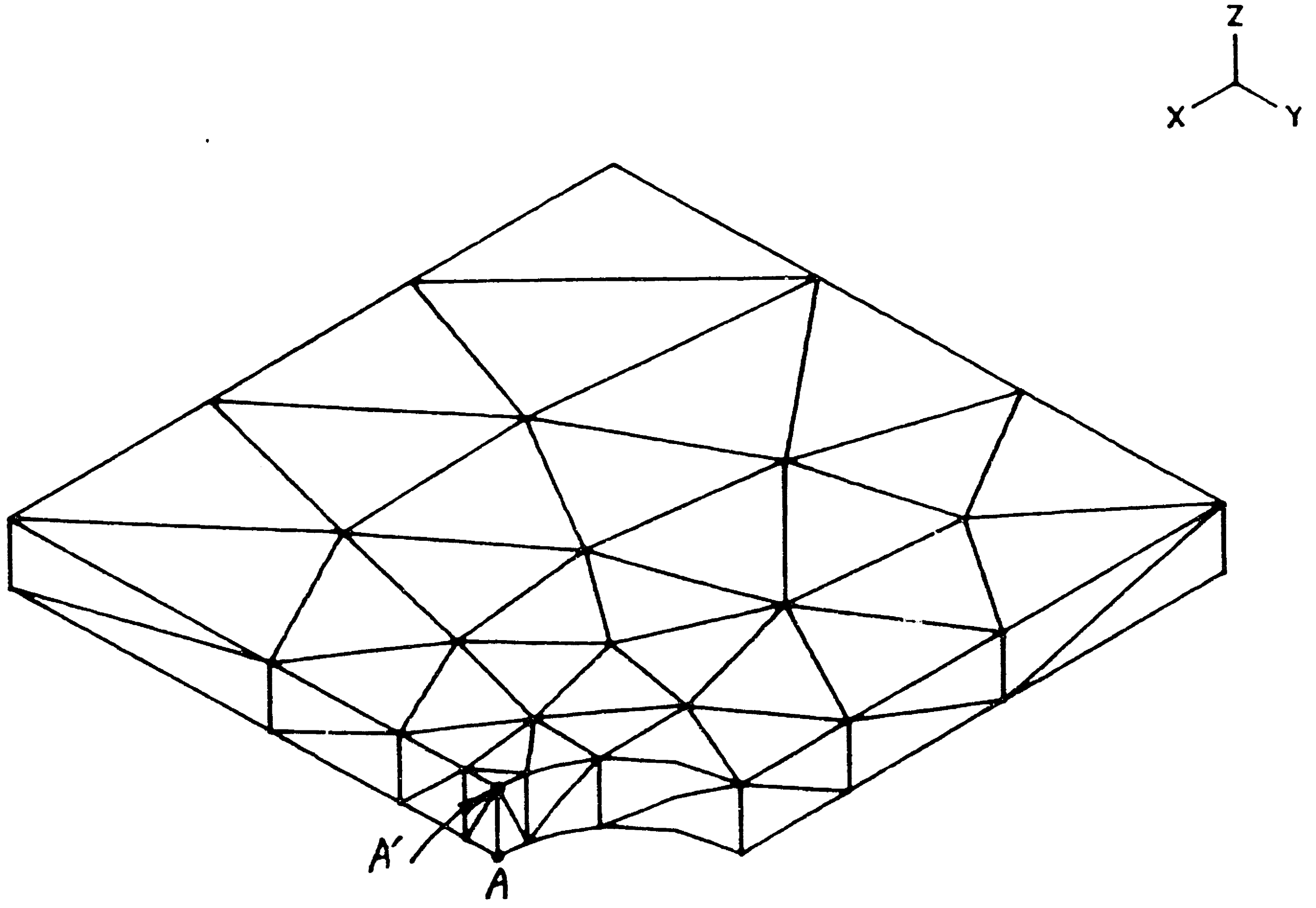
Figure 4-37: 39 element mesh with curved-sided elements



Total strain energy : 0.266 x 10^4 N-m

$\sigma_{xx}\big|_{A-A'} \cong 278$ MPa

Figure 4-38: 180 element mesh with curved-sided elements



Total strain energy : 0.267 x 10^4 N-m

$\sigma_{xx}\big|_{A-A'} \cong 343$ MPa

Figure 4-39: 108 element mesh with curved-sided elements

Table 4-3. The stress concentration factors for the meshes with curved-sided elements.

	Strain Energy N-m	Average Stress Concentration Factor along A-A'
39 element (uniform)	0.264×10^{4}	2.27
180 element (uniform)	0.266×10^{4}	2.78
108 element (graded)	0.267×10^{4}	3.43
Reference Value (2-D plane stress condition)		3.4

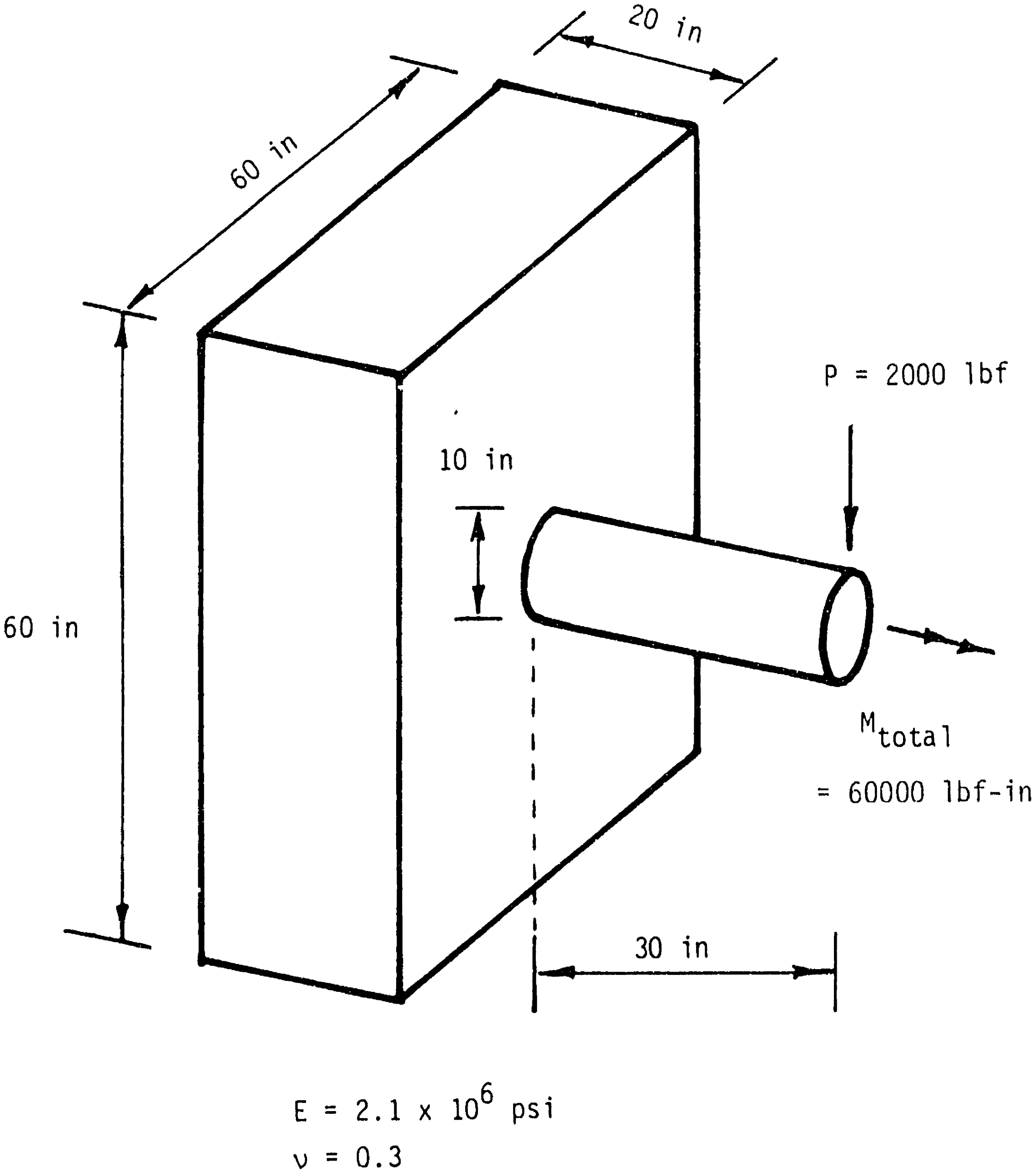


Figure 4-40: Cylinder attached to a block

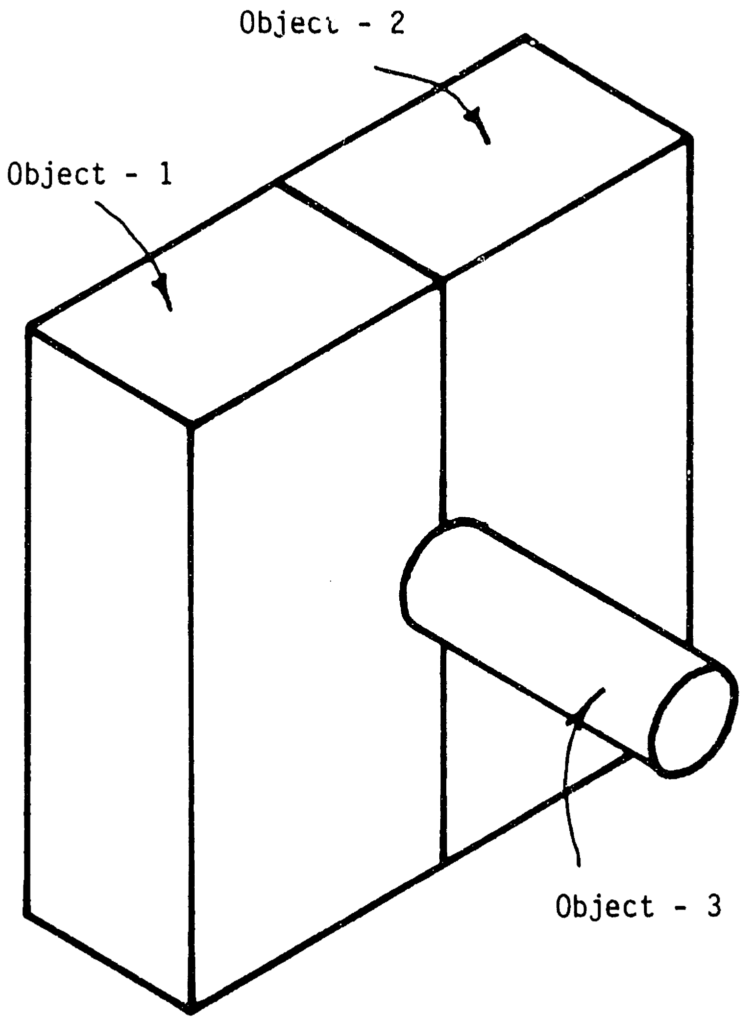


Figure 4-41: Subdivisions for a coarse mesh construction

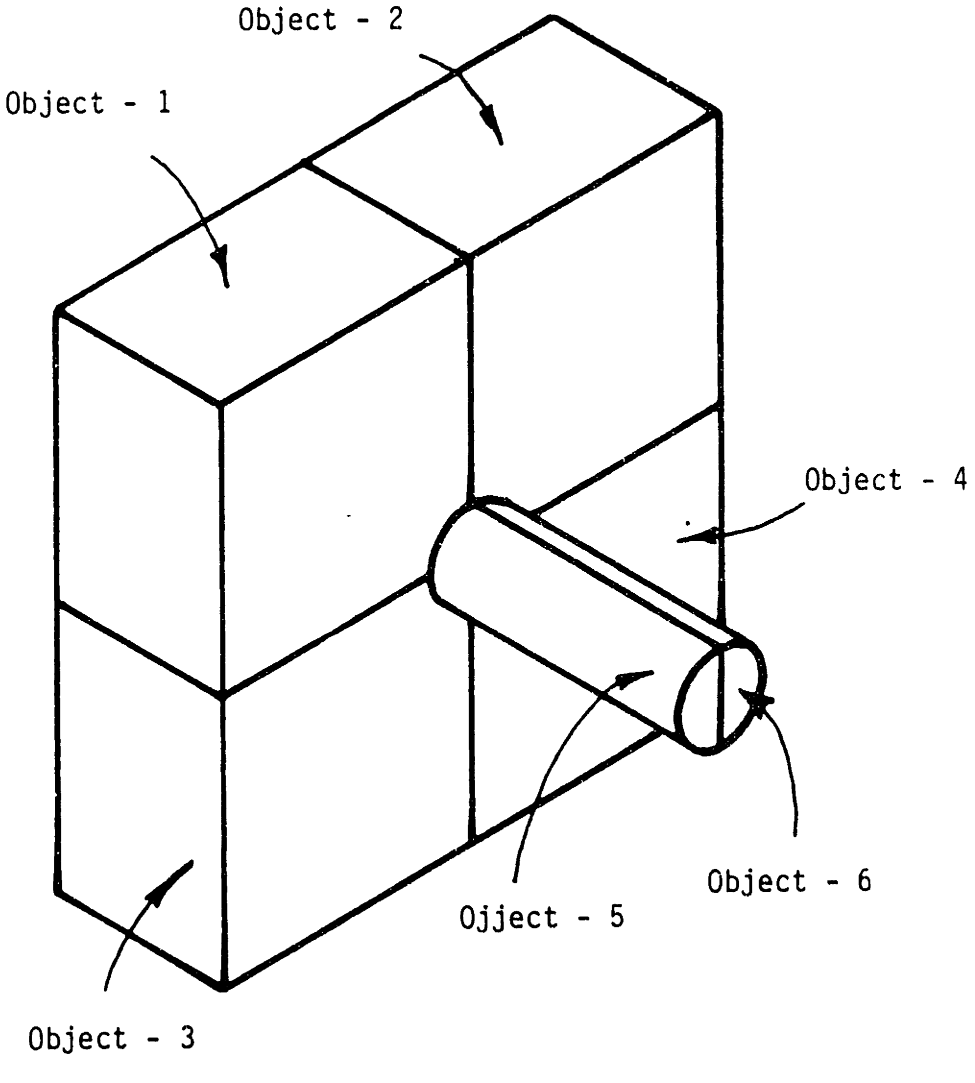
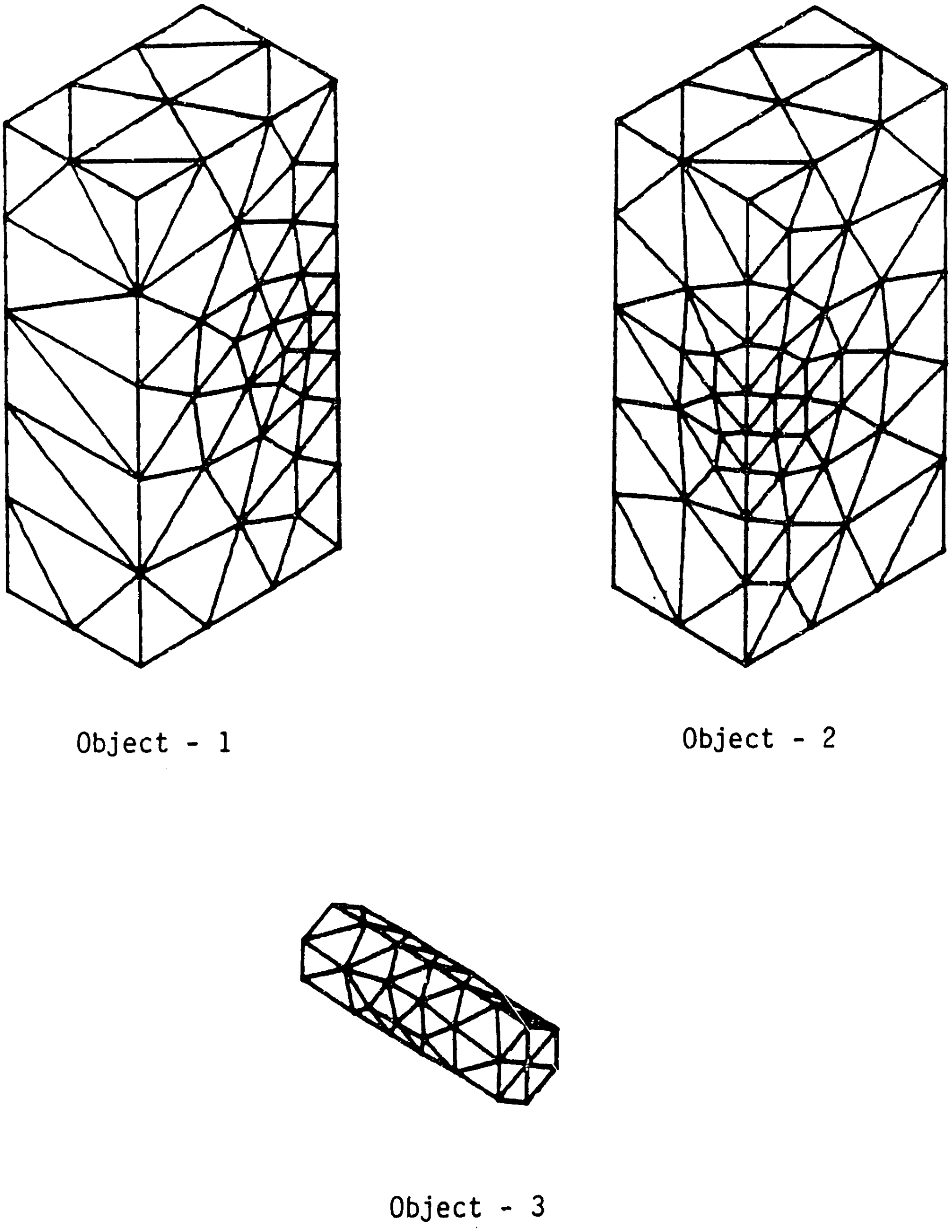


Figure 4-42: Subdivisions for a fine mesh construction



Object - 1

Object - 2

Object - 3

Figure 4-43 a): Detail of mesh generation for 514 element model

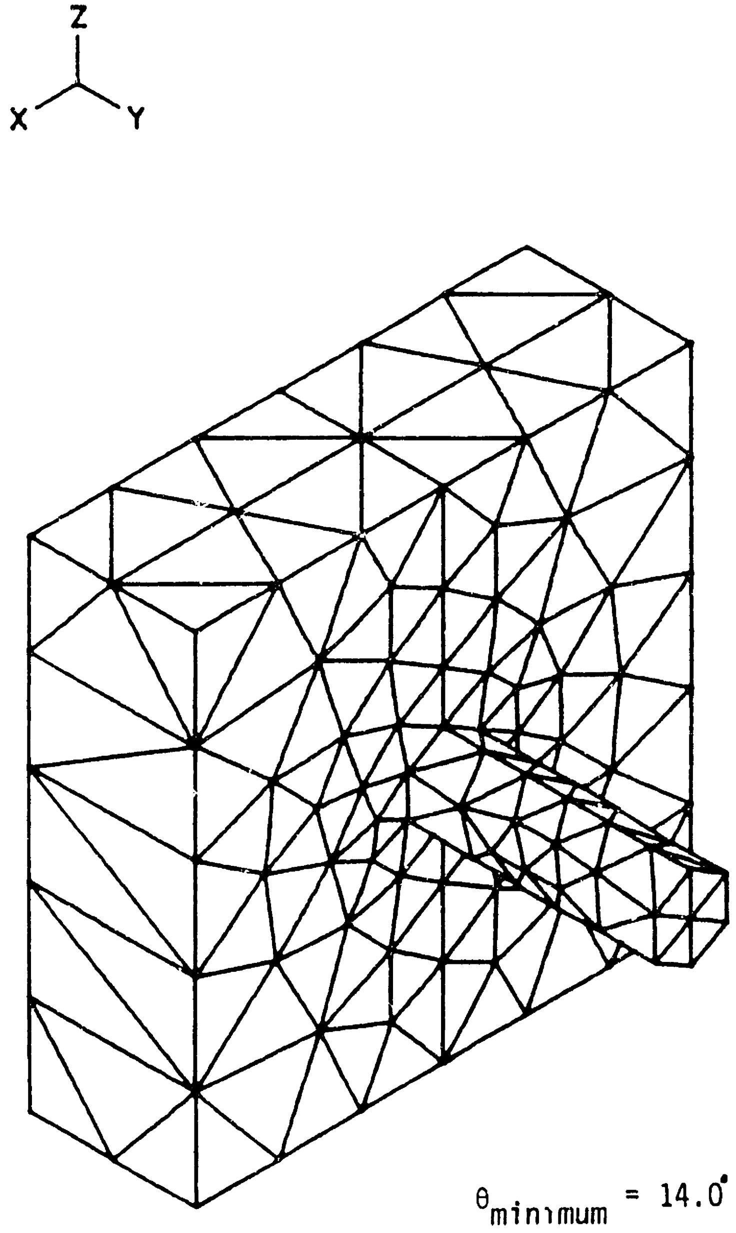
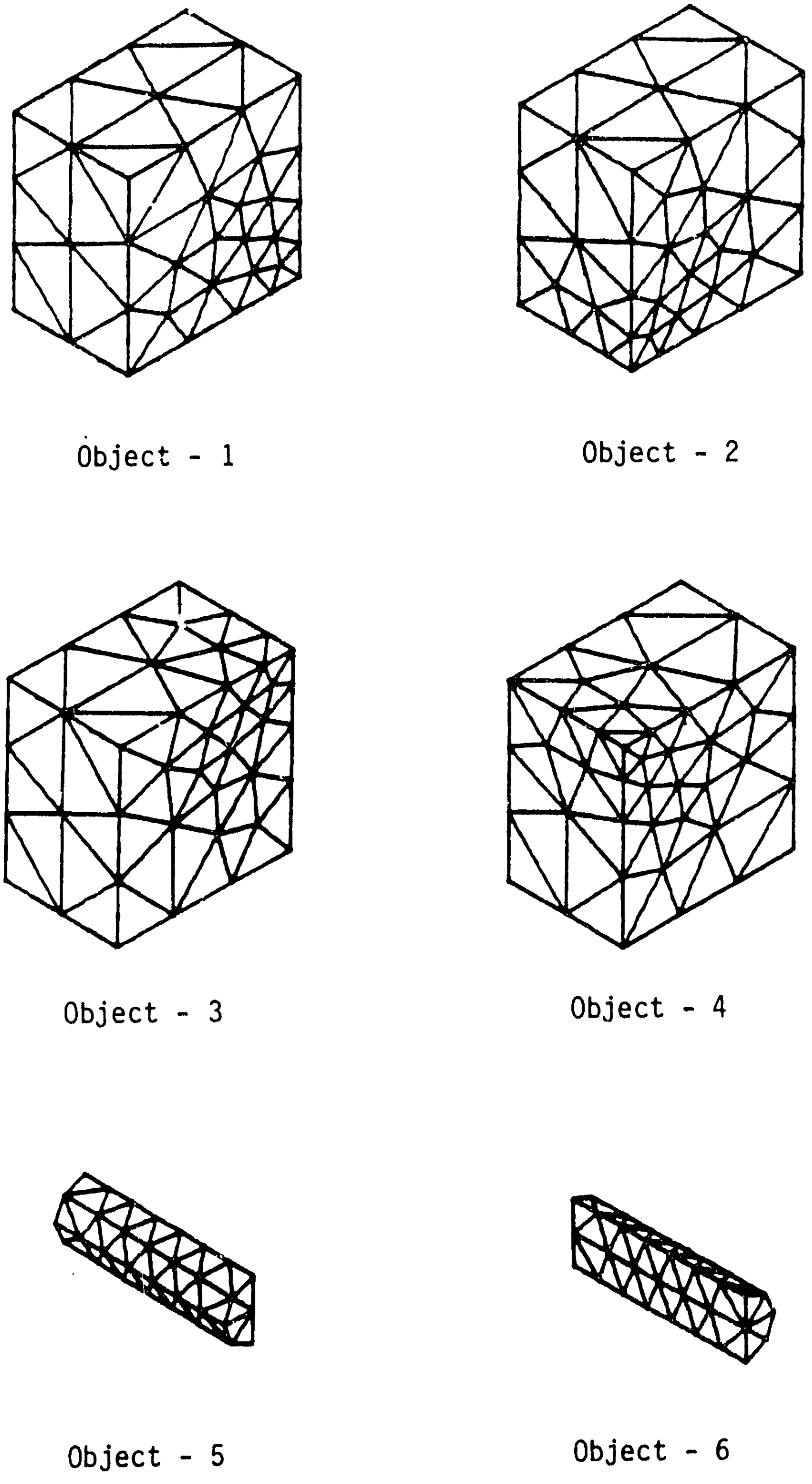


Figure 4-43 b): 514 element mesh with straight-sided elements



Object - 1

Object - 2

Object - 3

Object - 4

Object - 5

Object - 6

Figure 4-44 a): Detail of mesh generation for 1003 element model

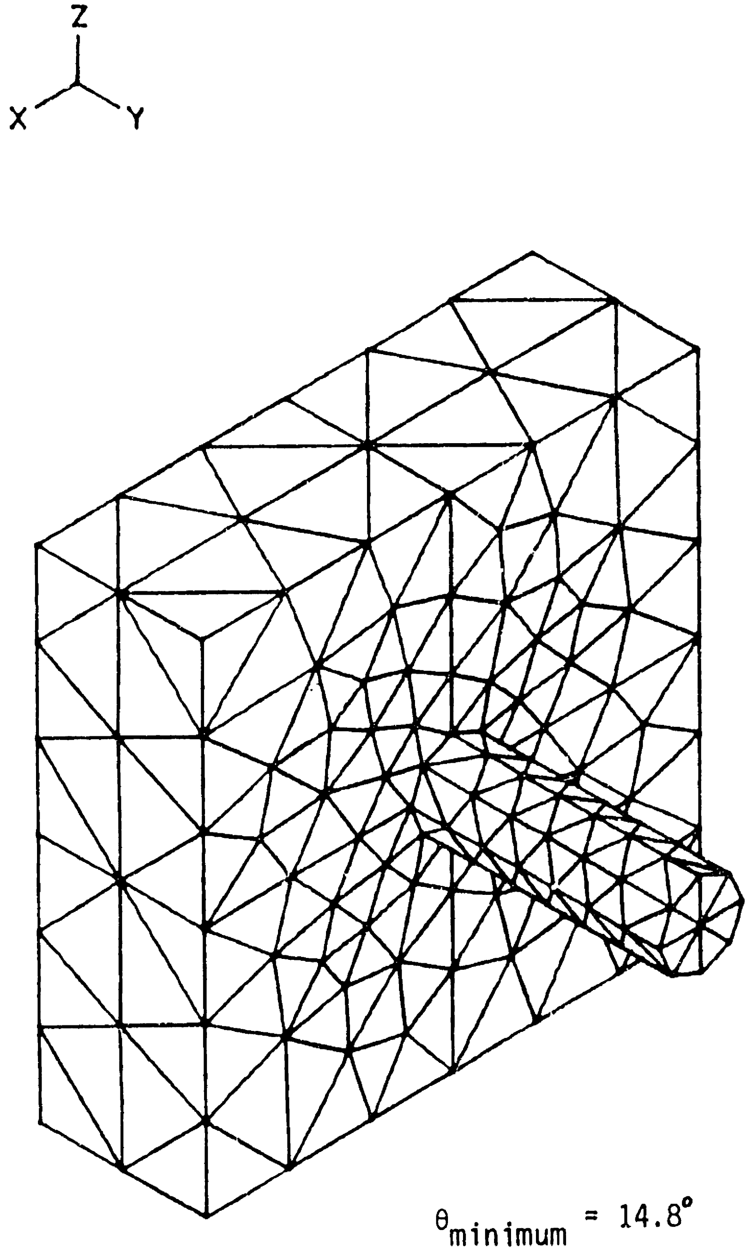
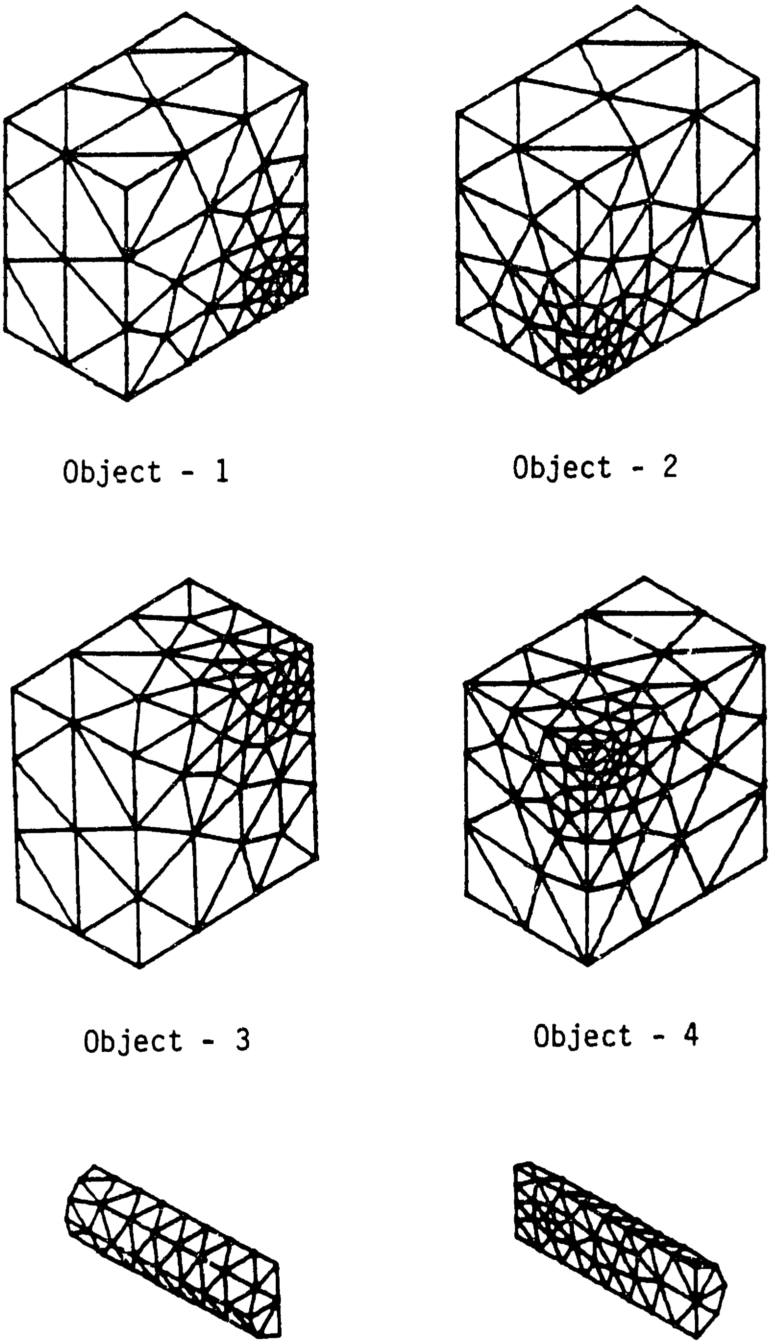


Figure 4-44 b): 1003 element mesh with straight-sided elements



Object - 1

Object - 2

Object - 3

Object - 4

Object - 5

Object - 6

Figure 4-45 a): Detail of mesh generation for 1568 element model

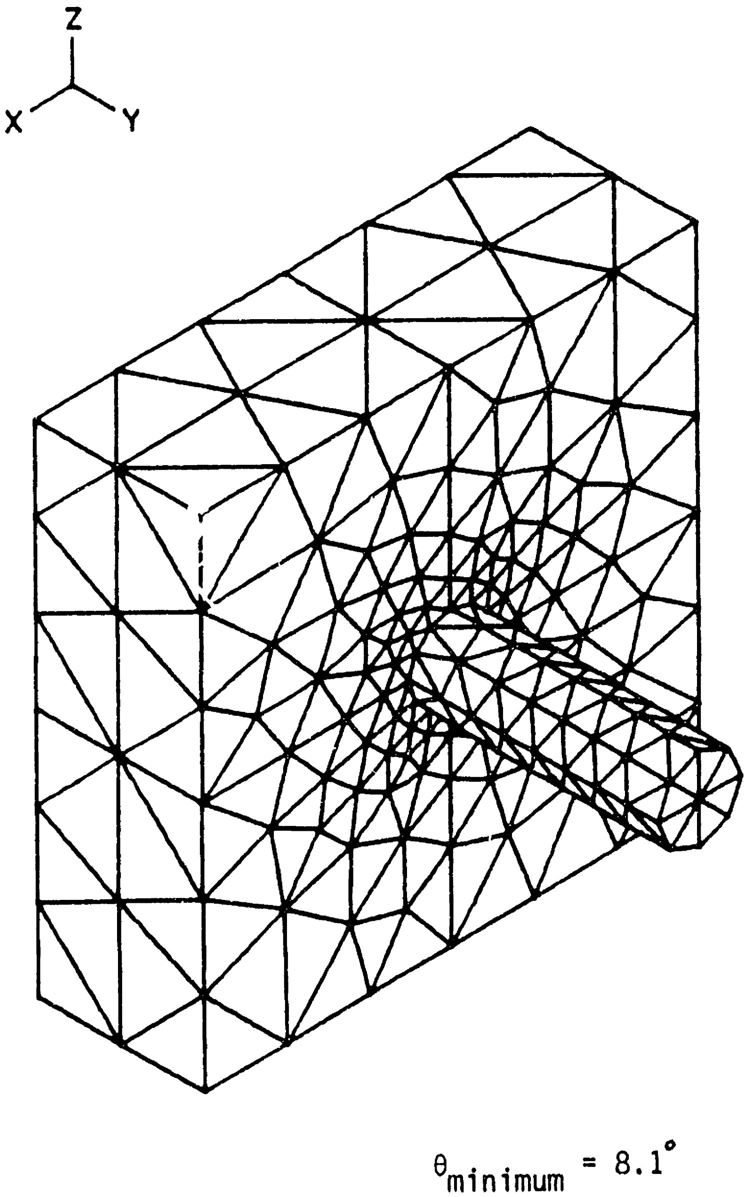


Figure 4-45 b): 1568 element mesh with straight-sided elements

other. However, the surface triangulation process does not generate exactly symmetric triangles on the surface due to the directionality involved in the algorithm, which will affect the volume triangulation process. Hence, the resulting meshes in object-1 and object-2 are different from each other and the same phenomena occur in the cases in Figure 4-44 and Figure 4-45.

As shown in Figure 4-43 to Figure 4-45, our mesh generation scheme works well, although in the case of the 1568 element mesh the minimum dihedral angle is $\theta_{min} = 8.1°$, which indicates that highly distorted elements are also generated.

The bending analysis for the meshes are performed as shown in Figure 4-46 and the results are summarized in Table 4-4. Notice that the total strain energy in the system decreases as more elements are employed, which contradicts finite element theory. The reason is due to the difference in the volume of the cylinder modeled by the different number of elements. Since only straight-sided elements are employed, the mesh with more elements in the cylinder results in a larger volume. Therefore, the stiffness of a fine mesh with 1568 elements is the largest among the three, which results in the smallest displacement at the tip.

Therefore, in a strict sense, the results are not directly comparable with each other. However, the stress outputs of the analyses compare reasonably with those of simple beam theory.

The results of the error analyses are shown in Figure 4-47. In the case of the 514 element mesh and the 1003 element mesh, the largest errors occur in the regions A, B, and C - in this order - where the mesh refinement is necessary. On the other hand, the largest errors in the 1568 element mesh occur in the regions A,

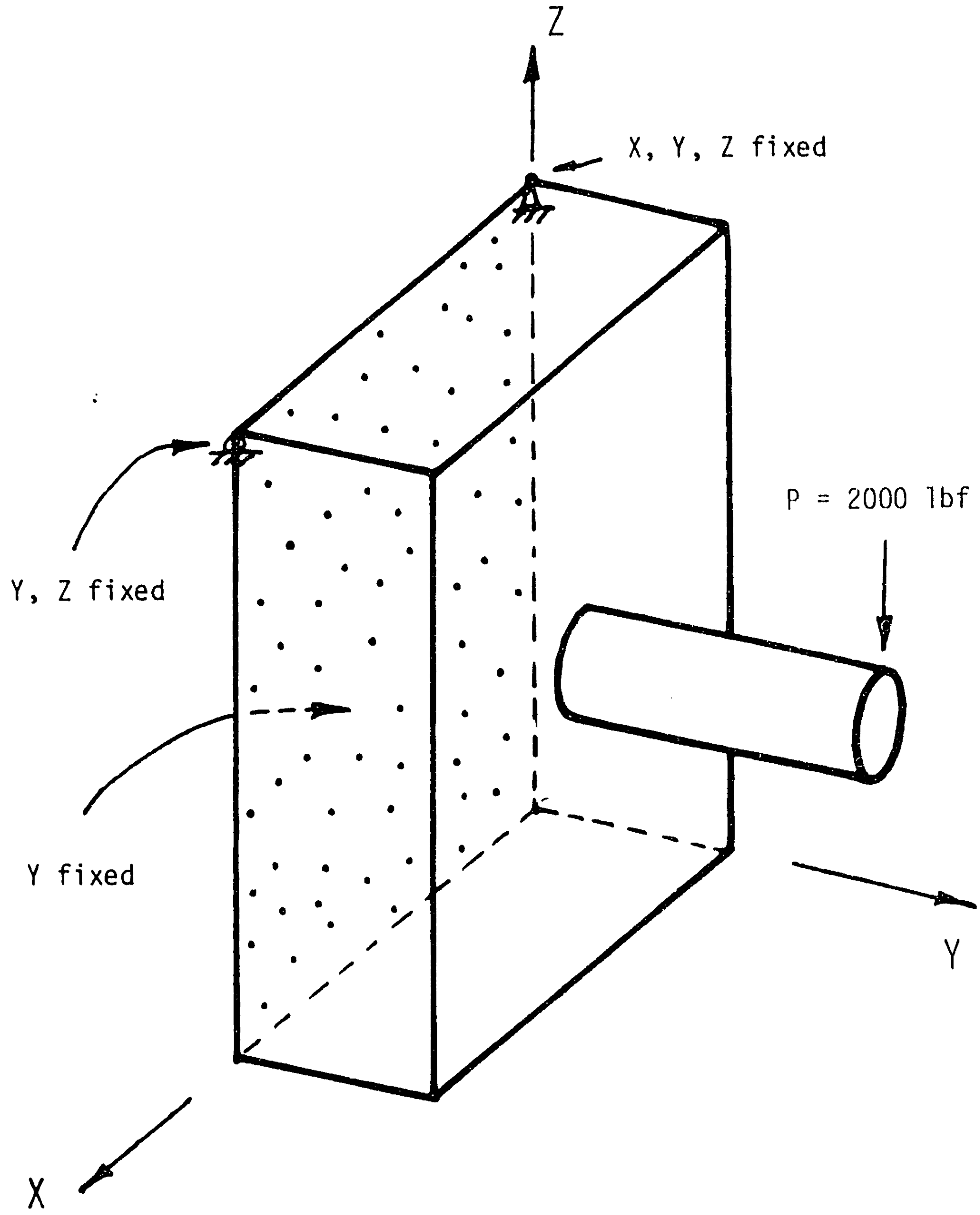
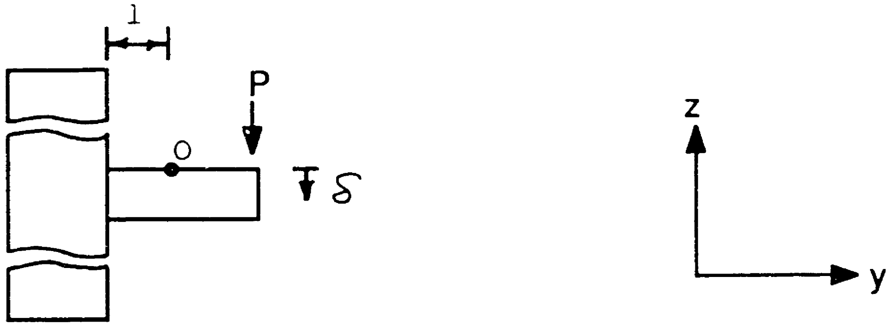
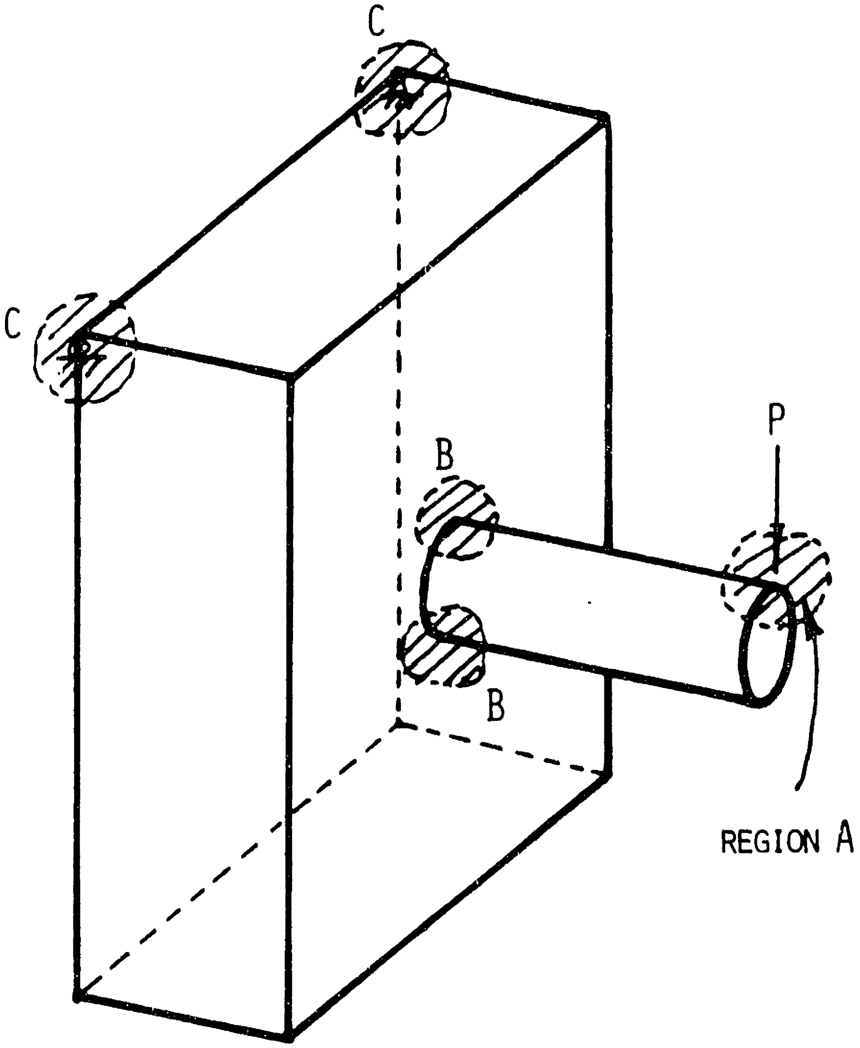


Figure 4-46: Loading conditions for bending analysis

Table 4-4. Results of the bending analysis.
(mesh with straight-sided elements)



Total No. of elements (No. of element in cylinder)	Volume in cylinder, (in ^{3})	Total strain energy (lbf- in)	Displacement δ (in)	Distance of O ℓ (in)	Average of σ_{yy} at O, (psi) % deviation (beam theory)	Total error indicator ε_{total}
514 elements (194 el.)	2021.4	31.37	-0.03137	10	548.5 34.6% (407.5)	55.87
1003 elements (259 el.)	2121.3	28.81	-0.02881	8.57	537.4 23.1% (436.6)	56.96
1568 elements (322 el.)	2134.2	28.63	-0.02863	7.5	564.3 23.1% (458.4)	58.37
Exact	2356					
Beam bending theory			-0.0175			



Largest errors obtained in the order :

	1st	2nd	3rd
514 elements;	A	B	C
1003 elements;	A	B	C
1568 elements;	A	C	B

Figure 4-47: Error distribution in the bending analysis

C, and B - in this new order - due to the small size of elements employed at the region B.

Since the tetrahedral element with only planar faces cannot accurately describe the curved surfaces of the cylinder, the tetrahedral element with curved faces should be employed for general three-dimensional analysis.

Hence the meshes considered already above are constructed again, but with curved sides in the tetrahedral elements to model the cylinder more accurately as shown in Figure 4-48, Figure 4-49, and Figure 4-50, respectively. The same bending analyses are performed for these meshes and the results are summarized in Table 4-5. The error analyses are not performed because only the element with planar faces is considered in our error analysis. The results in Table 4-5 show that the total strain energy increases as more elements are used, and the stress output at node O compares well with the one by beam theory. Therefore, using the elements with curved faces in the cylinder gives better results.

For the torsional analysis, the system under a twisting moment at one end of the cylinder as shown in Figure 4-51 is considered. In order to simulate the twisting moment, equivalent nodal forces are applied at one end of the cylinder, see the Figure 4-51. The equivalent nodal forces are obtained for the mesh with curved-sided elements, and thus are not exactly equivalent forces in the mesh with straight-sided elements.

The torsional analyses for the meshes with straight-sided elements in Figure 4-43 to Figure 4-45 are performed and the results are summarized in Table 4-6. In Table 4-6, we see again the contradictory results that the total strain energy in the system decreases as more elements are employed. The reason is also due to the difference in the volume of the cylinder.

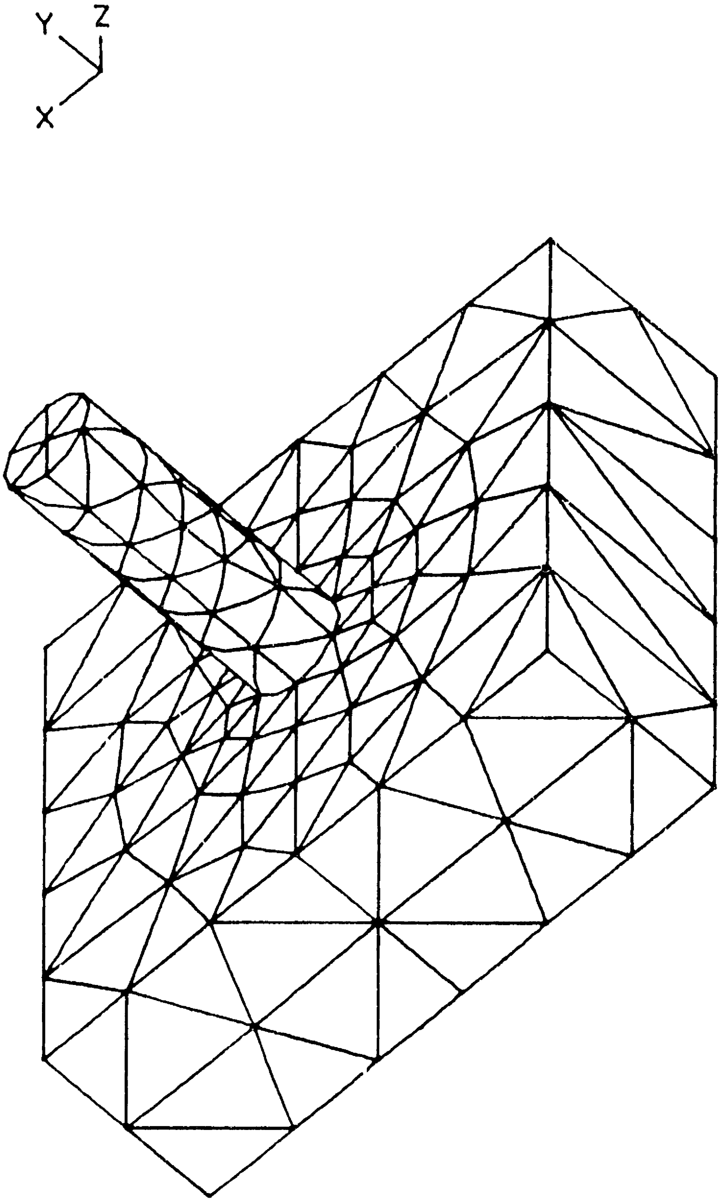


Figure 4-48: 514 element mesh with curved-sided elements

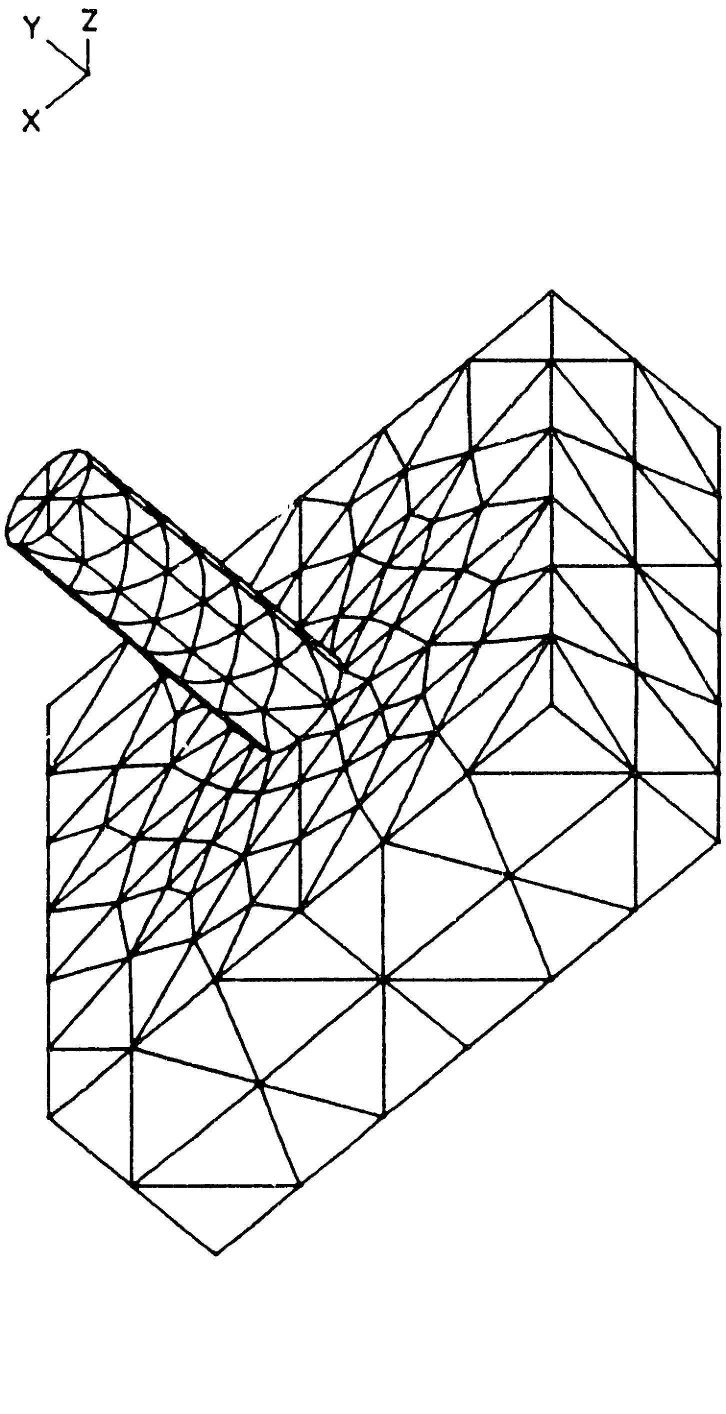


Figure 4-49: 1003 element mesh with curved-sided elements

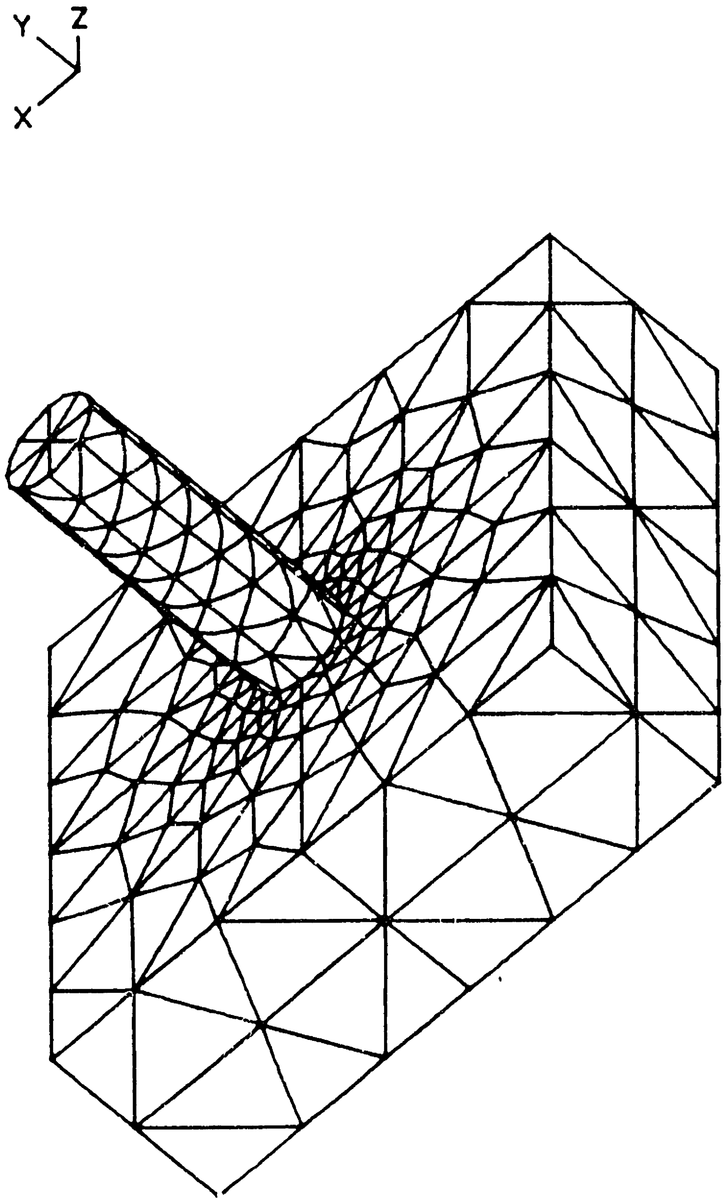
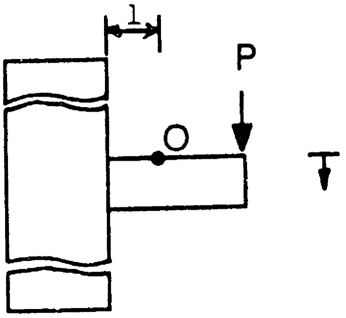


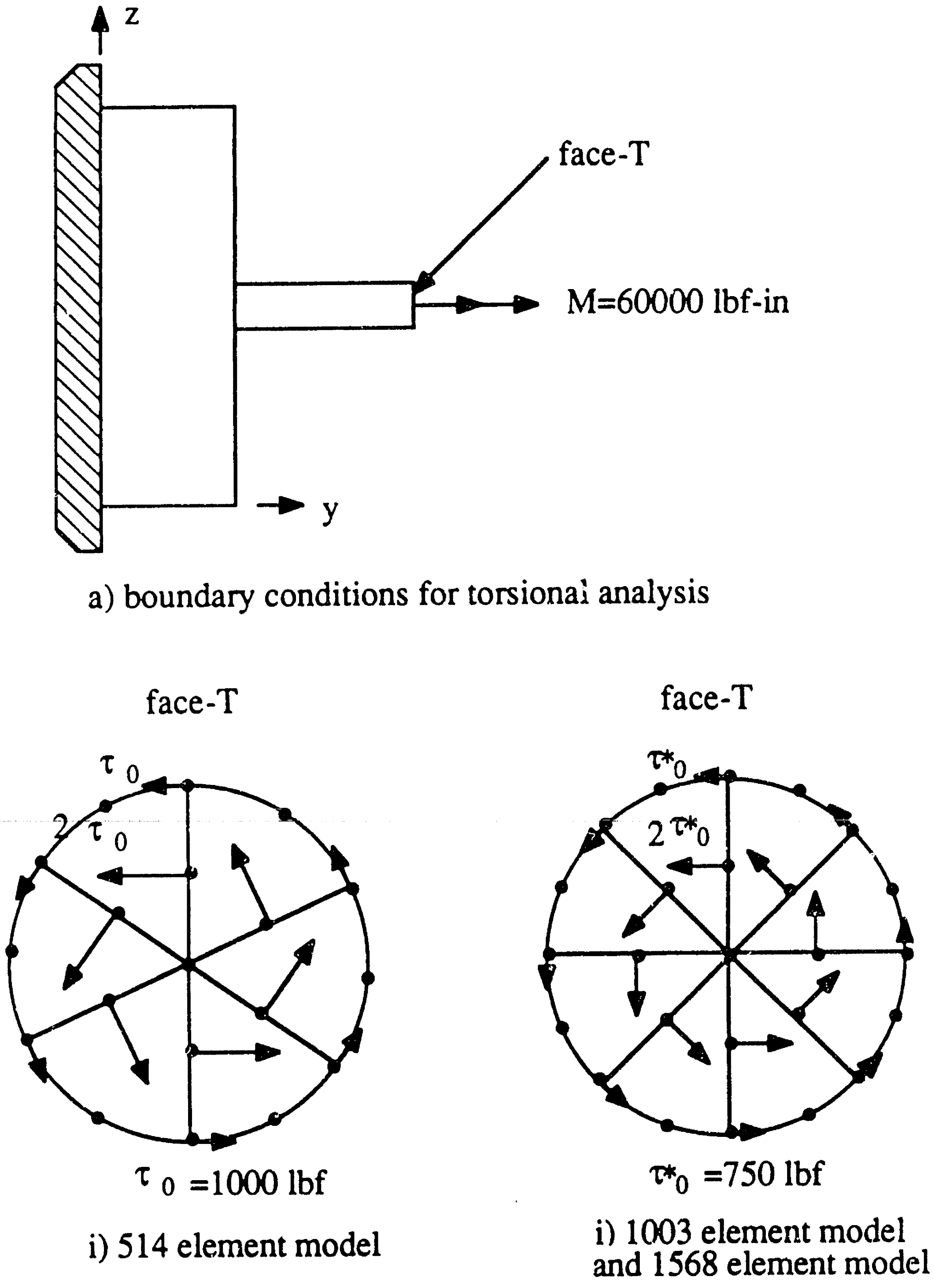
Figure 4-50: 1568 element mesh with curved-sided elements

Table 4-5. Results of the bending analysis (mesh with curved-sided elements)



δ ; displacement at the tip

Total No. of elements (No. of element in cylinder)	Total strain energy (lbf- in)	Displacement δ (in)	Distance of O l (in)	Average of σ_{yy} at O, %deviation (beam theory)
514 elements (194 el.)	23.3	-0.233x10^{-1}	10	429.0 5.3% (407.5)
1003 elements (259 el.)	23.9	-0.239x10^{-1}	8.57	459.4 5.2% (436.6)
1568 elements (322 el.)	24.6	-0.246x10^{-1}	7.5	480.4 4.8% (458.4)
Beam bending theory (cylinder only)		-0.175x10^{-1}		



a) boundary conditions for torsional analysis

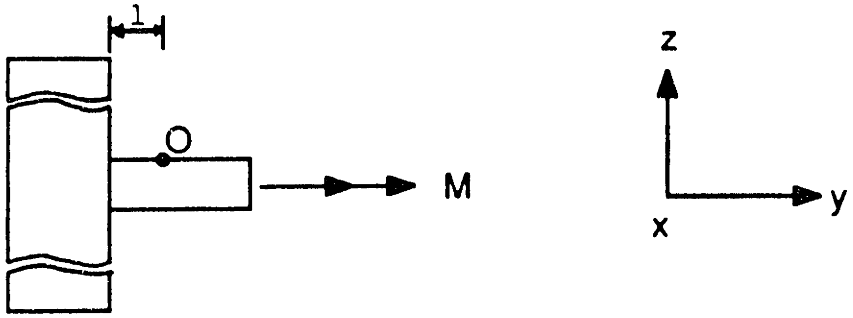
i) 514 element model

i) 1003 element model
and 1568 element model

b) Equivalent nodal forces

Figure 4-51: Loading conditions for torsional analysis

Table 4-6. Results of the torsional analysis (mesh with straight-sided elements)



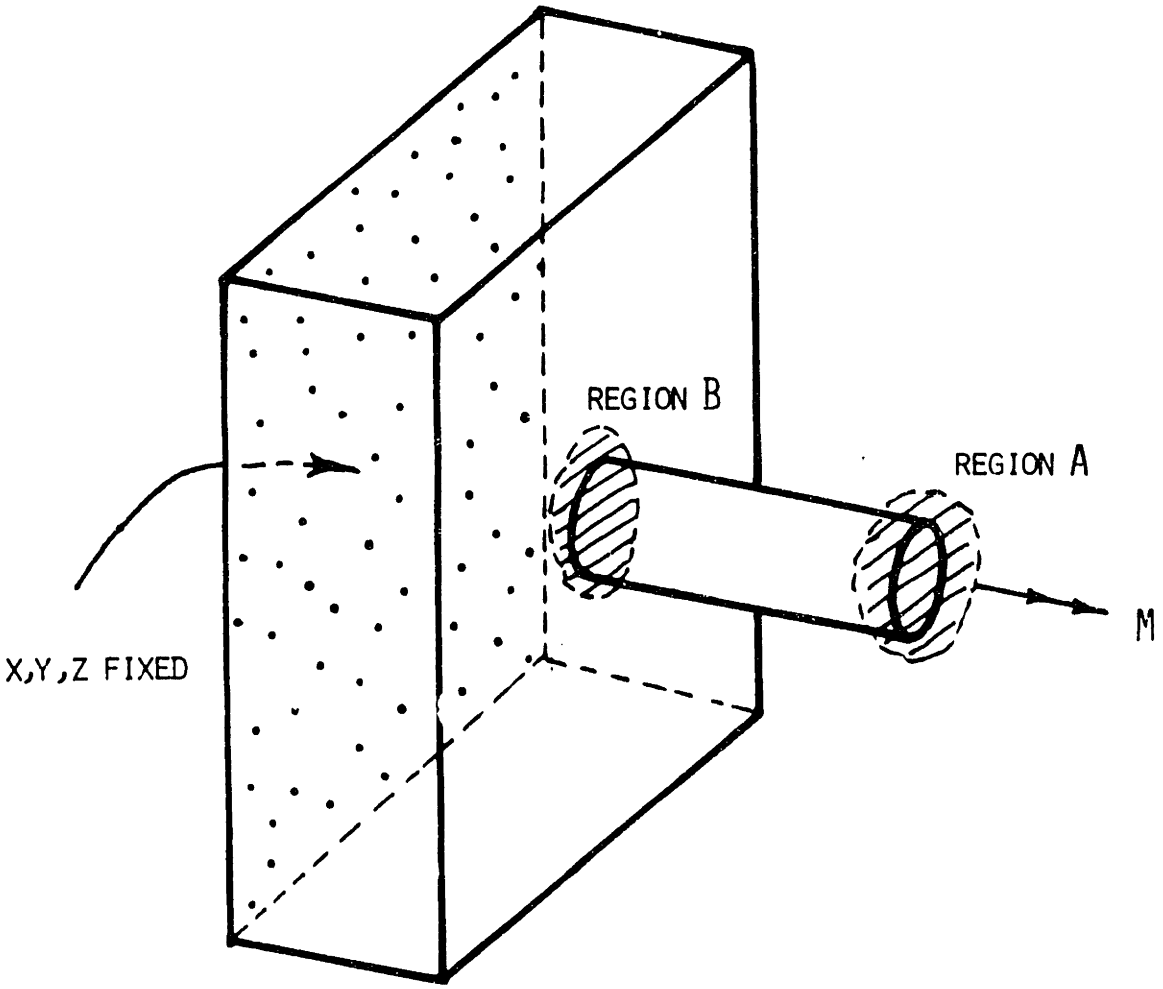
Total No. of elements (No. of element in cylinder)	Volume in cylinder (in^3)	Total strain energy (lbf ·in)	Total error indicator ε_{total}	Distance of O l (in)	Average of τ_{xy} at O (psi) (%deviation)
514 elements (194 el.)	2021.4	96.83	12.98	10	399.8 (30.8%)
1003 elements (259 el.)	2121.3	87.80	13.81	8.57	377.9 (23.6%)
1568 elements (322 el.)	2134.2	87.08	14.54	7.5	378.9 (24%)
Exact	2356				
Beam torsion theory					305.6

The results of error analyses are shown in Figure 4-52, in which the largest errors occur in the regions A and B , in this order. The largest error in the region A might be due to the applied loads in this region, which are not equivalent forces for the mesh with straight-sided elements. However, the large error in region B shows where refinement is necessary.

In order to obtain better results, the mesh with curved-sided elements in Figure 4-48 to Figure 4-50 are employed for the same torsional analyses and the results are summarized in Table 4-7. As shown in Table 4-7, the total strain energy in the system increases as more elements are employed and the stress outputs at node O compare well with the one by the beam torsion theory. Therefore, by using the mesh with curved-sided elements, reasonable results are obtained.

From the above results, we see that for general three-dimensional application the tetrahedral element which includes curved faces should be employed. In addition, we see that the error indicators used in our scheme give reasonable results and are applicable for the adaptive refinement process in the future. Although the error analysis for the element with curved faces is not included in our discussion, it can be implemented by using the procedures described in section 3.5.

Since the tetrahedral elements with curved faces are only used at the curved surface of the original system, the number of curved elements is relatively small compared with the total number of elements in the system. Therefore, it is probably best to implement two different procedures for the calculation of the same error indicator : one for elements with planar faces and the other for elements with curved faces.

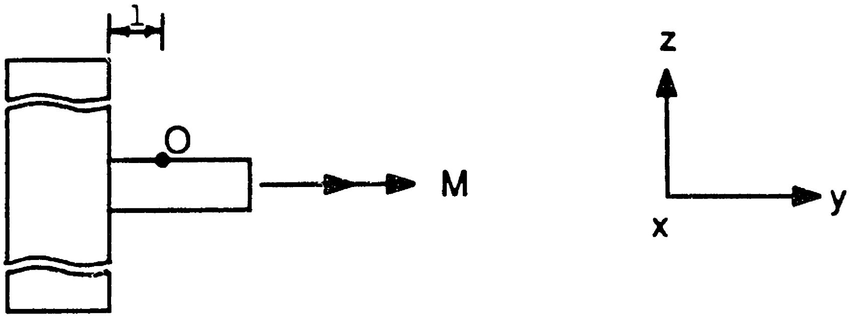


Largest errors obtained in the order :

	1st	2nd
514 elements;	A	B
1003 elements;	A	B
1568 elements;	A	B

Figure 4-52: Error distribution in the torsional analysis

Table 4-7. Results of the torsional analysis.
(mesh with curved-sided elements)



Total No. of elements (No. of elements in cylinder)	Total strain energy (lbf - in)	Distance of O l (in)	Average of τ_{xy} at O (psi) %deviation (beam theory)
514 elements (194 el.)	71.55	10	320.6 4.9% (305.6)
1003 elemunts (259 el.)	71.75	8.57	319.5 4.6% (305.6)
1568 elements (322 el.)	72.16	7.5	319.1 4.4% (305.6)

Chapter 5

Conclusions and Recommendations

We have presented a unified approach to the automatic generation of near-optimal finite element meshes both for two-dimensional and three-dimensional analysis. The proposed approach is composed of an efficient initial mesh construction and the *h-version* of self-adaptive refinement process.

For two-dimensional analysis, the whole procedures from an initial mesh construction to the self-adaptive refinement have been developed, while for three-dimensional analysis only an initial mesh construction and the error analysis are included. The self-adaptive refinement process for three-dimensional analysis is not included in the thesis. However, three-dimensional works conducted in the thesis provide the basis for self-adaptive refinement scheme as was proposed for two-dimensional analysis.

For an initial mesh construction, in two-dimensional analysis, a robust triangulation scheme has been developed, in which triangular elements are generated from the boundaries based on the key nodes which the user has to input.

With the suggested triangulation scheme, we are able to construct meshes with well-conditioned elements, unless we assign unrealistic key node distributions. However, since the triangulation scheme is sensitive to the key node positions, we may somtimes need to construct the mesh several times by changing the key node positions in order to obtain a better mesh. But this is not a serious restriction, because the amount of computation involved in the triangulation process is usually very small compared with that in the finite element analysis.

As a criterion for further refinement, an error indicator which employs the body force residuals in the energy norm with a relaxation factor is found to be most efficient for meshes with quadratic elements because better solutions can be obtained at stress concentrations with little sacrifice on the overall accuracy. The direct relationship between the actual accuracy of solution and a given tolerance of an error indicator is not identified in this research. Instead the error indicators computed are used as references for further refinement and the total error indicator indicates the error involved in the solution indirectly.

The adaptive refinement process for 8-node quadrilateral element is also found to be useful in obtaining a near-optimal mesh, if an initial mesh is prepared manually.

In the case of three-dimensional analysis, a new volume triangulation scheme is developed. The new scheme is obtained by employing the same concept of the surface triangulation scheme to three-dimensional problems. In this scheme, the user has to place the key nodes on the edges of an object and the program proceeds triangulation of the surfaces and the tetrahedronization of the volumes. The amount of computation involved in the volume triangulation is found to be very small compared with the one required for the finite element analysis. The topological and geometrical properties of the basic operations are examined in order to ensure that the algorithm converges.

The suggested volume triangulation scheme works reasonably well for a rather coarse mesh construction, while for a fine mesh construction some ill-conditioned elements may be generated. The reasons are largely due to the heuristic rules and the check processings employed in this scheme. Therefore further improvements will be necessary in the algorithm in order to obtain meshes with well-conditioned elements even for fine meshes.

Due to the directionality involved in the surface triangulation, which affects the volume triangulation process, the resulting meshes for symmetric subobjects may not be symmetric. Therefore, the capability of copying the resulting mesh of one part to other symmetric parts might be useful in the future.

In the example solutions persented, we see that curved-sided elements should be used both for two-dimensional and three-dimensional analysis in order to obtain reliable results. The error analysis for the tetrahedral element with curved-sided faces should be included as is described in section 3-5 and the self-adaptive refinement process should be included in the future. In implementing the error indicator, it is probably best to employ two different procedures for the same error indicator : one for elements with straight-sided faces and the other with curved faces.

In order to cope with the initial mesh construction through a solid modeler and the refinement process automatically, a special data structure, named "*dissembled winged-edge/face data structure*" is developed. The new data structure has a winged-edge like data structure for two-dimensional problems and has a winged-face like data structure for three-dimensional problems and the original object/loop is considered to be an assemblage of many subobjects/subloops. The complexity involved in updating the connectivity information during the refinement process can be overcome by employing pre-designed refinement units. By using this boundary representation data structure, the data management can be easily handled without extensive use of searches.

All the above procedures consider only systems with straight lines and circular arcs in the two-dimensional case, and planar and cylindrical surfaces in the three-dimensional case. For general applications, additional curves and surfaces should also be included in the future. For two-dimensional applications, additional curves may be included without much difficulty, but for three-dimensional applications it is still a challenge to include other surfaces, such as quadratic or composite surfaces ,due to the difficulties involved in the triangulation of these surfaces.

In a practical CAD environment, a complex object is frequently input by digitizing the nodes on the surfaces. In this case, if the surfaces of an object are triangulated manually and the results are input by a digitizer, the suggested volume triangulation scheme can also be used for the mesh generation. This is another possibility for the practical application of our volume triangulation scheme.

References

[1] G.M. McNeice and P.V. Marcal.
Optimization of Finite Element Grids Based on Minimum Potential Energy.
Trans. ASME. J. of Engineering for Industry 95(1):186-190, 1973.

[2] D.J. Turcke and G.M. McNeice.
Guidlines for Selecting Finite Element Grids Based on an Optimization Study.
Computers and Structures 4:499-519, 1974.

[3] A.R. Diaz, N. Kikuchi and J.E. Taylor.
A Method of Grid Optimization for Finite Element Methods.
Computer Methods in Applied Mechanics and Engineering 41:29-45, 1983.

[4] T. Sussman and K.J. Bathe.
The Gradient of the Finite Element Variational Indicator with Respect to Nodal Point Coordinates: An Explicit Calculation and Applications in Fracture Mechanics and Mesh Optimization.
Int. J. Num. Methods in Engineering 21:763-774, 1985.

[5] I. Babuska and W.C. Rheinboldt.
Error Estimates for Adaptive Finite Element Computations.
SIAM J. Numerical Analysis 15:736-754, 1978.

[6] I. Babuska and W.C. Rheinboldt.
Adaptive Approaches and Reliability Estimations in Finite Element Analysis.
Computer Methods Appl. Mech. Engineering 17/18:519-540, 1979.

[7] I. Babuska and W.C. Rheinboldt.
A Posteriori Error Estimates for the Finite Element Method.
Int. J. Numer. Methods in Engineering 112:1597-1615, 1978.

[8] D.W. Kelly, J.P.De S.R. Gago, O.C. Zienkiewicz and I. Babuska.
A Posteriori Error Analysis and Adaptive Processes in the Finite Element Method: Part I - Error Analysis.
Int. J. Numer. Methods Engineering 19:1593-1619, 1983.

[9] J.P.De S.R. Gago, D.W. Kelly, O.C. Zienkiewicz and I.Babuska.
A Posteriori Error Analysis and Adaptive Processes in the Finite Element Method: Part II - Adaptive Mesh Refinement.
Int. J. Numer. Methods Engineering 19:1621-1656, 1983.

[10] T. Itoh.
Adaptive Finite Element Method in Two-Dimensional Structural Problems.
PhD thesis, Univ. of Califomia, Berkeley, 1981.

[11] I. Babuska and Dehao Yu.
Asymptotically Exact A-Posteriori Error Estimator for Biquadratic Elements.
Technical Report BN-1050, Institute for Physical Science and Technology, University of Maryland, June, 1986.

[12] O.C. Zienkiewicz and J.Z. Zhu.
A Simple Error Estimator and Adaptive Procedure for Practical Engineering Analysis.
Int. J. Numer. Methods in Engineering 24:337-357, 1987.

[13] Graham F. Carey and J. Tinsley Oden.
Finite Elements: Computational Aspects.
Prentice-Hall, Inc., Englewood Cliffs, 1984.

[14] I. Babuska, O.C. Zienkiewicz, J. Gago and E.R.de A. Oliveira.
A Wiley-Interscience Publication: *Accuracy Estimates and Adaptive Refinements in Finite Element Computations.*
John Wiley & Sons, 1986.

[15] O.C. Zienkiewicz, J.P. De S.R. Gago and D.W. Kelly.
The Hierarchical Concept in Finite Element Analysis.
Computers & Structures 16(1-4):53-65, 1983.

[16] O.C. Zienkiewicz, D.W. Kelly, J. Gao and I. Babuska.
Hierarchical Finite Element Approaches, Error Estimates and Adaptive Refinement.
Proc. MAFELAP :313-346, 1981, Brunel University.

[17] Bama A. Szabo.
Mesh Design for the p-Version of the Finite Element Method.
Computer Methods in Applied Mechanics and Engineering 55:181-197, 1986.

[18] D.J. Turcke.
ASME Special Publications. Number PVP-38: *Characteristics of Piecewise Approximations in Numerical Analysis.*
ASME, 1979.

[19] D.J. Turcke and G.M. McNeice.
Guidelines for Selecting Finite Element Grids Based on an Optimization Study.
Computers & Structures 4:499-519, 1974.

[20] M.S. Shephard, R.H. Gallagher and J.F. Abel.
The Synthesis of Near-Optimum Finite Element Meshes with Interactive Computer Graphics.
Int. J. Numer. Methods. Eng. 15:1021-1039, 1980.

[21] P. Ladeveze and D. Leguillon.
Error Estimate Procedure in the Finite Element Method and Applications.
SIAM J. Numer. Analysis 20(3):485-509, June, 1983.

[22] E.R.A. Oliveria.
Optimization of Finite Element Solutions.
Proc. 3rd Conf. on Matrix Methods in Structural Mechanics, Wright-Patterson Air Force Base , October, 1971.

[23] M.S. Shephard.
Finite Element Grid Optimization with Interactive Computer Graphics.
PhD thesis, Dept. of Structural Engr., School of Civil Engr., Cornell University, January, 1979.

[24] W.C. Thacker.
A Brief Review of Techniques for Generating Irregular Computational Grids.
Int. J. Numer. Methods Eng. 15(9):1335-1341, September, 1980.

[25] J.C. Cavendish.
Automatic Triangulation of Arbitrary Planar Domains for the Finite Element Method.
Int. J. Numer. Methods Eng. 8:679-696, 1974.

[26] R.D. Shaw and R.G. Pitchen.
Modifications to the Suhara-Fukuda Method of Network Generation.
Int. J. Numer. Methods Eng. 12:93-99, 1978.

[27] Ichiei Imafuku, Yoichi Kodera, Masaaki Sayawaki and Makoto Kono.
A Generalized Automatic Mesh Generation Scheme for Finite Element Method.
Int. J. Numer. Methods Eng. 15:713-731, 1980.

[28] S.H. Lo.
A New Mesh Generation Scheme for Arbitrary Planar Domains.
Int. J. Numer. Methods Eng. 21:1403-1426, 1985.

[29] A. Bykat.
Automatic Generation of Triangular Grid: I - Subdivision of a General Polygon into Convex Subregions; II - Triangulation of Convex Polygons.
Int. J. Numer. Methods Eng. 10:1329-1342, 1976.

[30] A.J.G. Schoofs, L.H.Th.M. van Beukering and M.L.C. Sluiter.
TRIQUAMESH. Gebruikershandleiding.
Technical Report The-rapport WE 78-01, Technische Hogeschool Eindhoven, Afdeling der Wertuigbouwkunde, 1978.

[31] M.A. Yerry and M.S. Shephard.
A Modified-Quadtree Approach to Finite Element Mesh Generation.
IEEE Computer Graphics and Applications 3(1):36-46, 1983.

[32] M.S. Shephard, N.A.B. Yehia, G.S. Burd and T.J. Weidner.
Automatic Crack Propagation Tracking.
Computers and Structures 20:211-223, 1985.

[33] M.S. Shephard and M.A. Yerry.
Approaching the Automatic Generation of Finite Element Meshes.
Computers in Mech. Engr. 1(4):49-56, 1983.

[34] F.T. Tracy.
Graphics Pre- and Post-Processor for Two-Dimensional Finite Element Programs.
Computer Graphics (Proc. Siggraph '77) 11(2):8-12, August, 1977.

[35] E.A. Sadek.
A Scheme for the Automatic Generation of Triangular Finite Elements.
Int. J. Numer. Methods Eng. 15(12):1813-1822, December, 1980.

[36] Masaaki Yokoyama.
Automated Computer Simulation of Two-Dimensional Elastostatic Problems by the Finite Element Method.
Int. J. Numer. Methods in Eng. 21:2273-2287, 1985.

[37] Mark Shephard.
Finite Element Modeling Within an Integrated Geometric Modeling Environment: Part I - Mesh Generation.
Engineering with Computers 1:61-71, 1985.

[38] J.C. Cavendish, D.A. Field and W.H. Frey.
An Approach to Automatic Three-Dimensional Finite Element Mesh Generator.
Int. J. Numer. Methods Eng. 21:329-347, 1985.

[39] V.Ph. Nguyen.
Automatic Mesh Generation with Tetrahedron Elements.
Int. J. Numer. Methods Eng. 18:273-280, 1982.

[40] M.L.C. Sluiter and D.C. Hansen.
A General Purpose Automatic Mesh Generator for Shell and Solid Finite Elements.
Comput. Eng. 3(G00217, Ed. L.E. Hulbert, ASME):29-34, 1982.

[41] M.A. Yerry and M.S. Shephard.
Automatic Three-Dimensional Mesh Generation by the Modified-Octree Technique.
Int. J. Numer. Methods Eng. 20:1965-1990, 1984.

[42] M.A. Yerry and M.S. Shephard.
Automatic Mesh Generation for Three-Dimensional Solids.
Computers and Structures 20:31-39, 1985.

[43] Mark S. Shephard, Mark A. Yerry and Peggy L. Baehmann.
Automatic Mesh Generation Allowing for Efficient a Priori and a Posteriori Mesh Refinement.
Computer Methods in Applied Mechanics and Engineering 55:161-180, 1986.

[44] T.C. Woo and T. Thomasma.
An Algorithm for Generating Solid Elements in Objects with Holes.
Computers and Structures 8(2):333-342, 1984.

[45] B. Wördenweber.
Finite Element Mesh Generation.
Computer-Aided Design 16(5):285-291, 1984.

[46] A.A.G. Requicha and H.B. Voelcker.
Solid Modeling: A Historical Summary and Contemporary Assessment.
IEEE Computer Graphics and Applications 2(2):9-24, March, 1982.

[47] Bruce G. Baumgart.
Geometric Modeling for Computer Vision.
PhD thesis, Stanford University, August, 1974.

[48] Kunwoo Lee.
Data Structure and Fortran Arrays for Assembly and Component.
Technical Report Doc. No. 39, M.I.T. CAD Laboratory, January, 1982.

[49] ADINA-IN.
A Program for Generation of Input Data to ADINA.
ADINA R&D Report ARD 84-6, December, 1984.

[50] ADINA.
A Finite Element Program for Automatic Dynamic Incremental Nonlinear analysis.
ADINA R&D Report ARD 84-1, December, 1984.

[51] O.C. Zienkiewicz and D.V. Phillips.
An Automatic Mesh Generation Scheme for Plane and Curved Surfaces by Isoparametric Co-ordinates.
Int. J. Numer. Methods Eng. 3:519-528, 1971.

[52] G. Strang and G.J. Fix.
An Analysis of the Finite Element Method.
Prentice-Hall, Inc., Englewood Cliffs, N.J., 1973.

[53] W.C. Rheinboldt and C.K. Mesztenyi.
On a Data Structure for Adaptive Finite Element Mesh Refinements.
ACM Transaction on Math. Software 6(2):166-187, June, 1980.

[54] ADINA.
ADINA Theory and Modeling Guide.
ADINA R&D Report ARD 84-4, December, 1984.

[55] I.D. Faux and M.J. Pratt.
Computaticnal Geometry for Design and Manufacture.
John Wiley & Sons, 1979.

[56] James D. Foley and Andries Van Dam.
Fundamentals of Interactive Computer Graphics.
Addison-Wesley Publishing Company, 1982.

[57] Terry L. Janssen.
A Simple Efficient Hidden Line algorithm.
Computers and Structures 17(4):563-571, 1983.

[58] Klaus-Jürgen Bathe.
Finite Element Procedures In Engineering Analysis.
Prentice-Hall, Inc., Englewood Cliffs, N.J., 1982.

[59] C.G. Floyd.
The Determination of Stresses Using a Combined Theoretical and Experimental Analysis Approach.
2nd Int. Conf. on Computational Methods and Experimental Measurements (Ed. C.A. Brebbia), 1984.

[60] T. Sussman and K.J. Bathe.
Studies of Finite Element Procedures - On Mesh Selection.
Computers and Structures 21(1/2):257-264, 1985.

[61] Theodore Sussman and Klaus-Jürgen Bathe.
Studies of Finite Element Procedures - Stress Band Plots and the Evaluation of Finite Element Meshes.
Engineering Computations 3:178-191, September, 1986.

[62] S.P. Timoshenko and J.N. Goodier.
Theory of Elasticity.
McGraw-Hill, 1970.

[63] ADINA.
ADINA Verification Manual.
ADINA R&D Report ARD 84-5, December, 1984.

[64] H. Tada, P.C. Paris and G.R. Irwin.
The Stress Analysis of Cracks Handbook.
Del Research Corp.,Hellertown, PENN., 1973.

[65] R.J. Roark and W.C. Young.
Formulas for Stress and Strain, 5th Edition.
McGraw-Hil., New York, 1975.

Appendix A
Relationship between $F_i^{\Omega_m}$ and $F_i^{\Gamma_m}$

For each finite element, the two sources of error are written as,

$$R_i^{FE} = \tau_{ij}^{FE}{}_{,j} + f_i^B \neq 0$$

$$T_i^{FE} = \tau_{ij}^{FE} n_j - t_i \neq 0$$

while the exact solution satisfies the following equilibrium equations

$$\tau_{ij,j} + f_i^B = 0$$

$$\tau_{ij} n_j - t_i = 0$$

In section 2.2.2 the body force residual error, $F_i^{\Omega_m}$ and the traction residual error, $F_i^{\Gamma_m}$ in the force unit are defined as

$$F_i^{\Omega_m} = \int_{\Omega_m} R_i^{FE} \, d\Omega_m$$

$$F_i^{\Gamma_m} = \int_{\Gamma_m} T_i^{FE} \, d\Gamma_m$$

By using the Gauss divergence theorem for the following terms such as,

$$\int_{\Omega_m} \tau_{ij}{}^{FE}{}_{,j} d\Omega_m = \int_{\Gamma_m} \tau_{ij}{}^{FE} n_j d\Gamma_m$$

$$\int_{\Gamma_m} \tau_{ij} n_j d\Gamma_m = \int_{\Omega_m} \tau_{ij,j} d\Omega_m$$

we obtain the following relationship between $F_i^{\Omega_m}$ and $F_i^{\Gamma_m}$

$$\begin{aligned}
F_i^{\Omega_m} &= \int_{\Omega_m} R_i^{FE} d\Omega_m = \int_{\Omega_m} (\tau_{ij}{}^{FE}{}_{,j} + f_i^B) d\Omega_m \\
&= \int_{\Gamma_m} \tau_{ij}{}^{FE} n_j \, d\Gamma_m + \int_{\Omega_m} f_i^B \, d\Omega_m \\
&= \int_{\Gamma_m} (\tau_{ij}{}^{FE} n_j - t_i) \, d\Gamma_m + \int_{\Gamma_m} t_i \, d\Gamma_m + \int_{\Omega_m} f_i^B \, d\Omega_m \\
&= F_i^{\Gamma_m} + \int_{\Gamma_m} \tau_{ij} n_j \, d\Gamma_m + \int_{\Omega_m} f_i^B \, d\Omega_m \\
&= F_i^{\Gamma_m} + \int_{\Omega_m} (\tau_{ij,j} + f_i^B) \, d\Omega_m \\
&= F_i^{\Gamma_m}
\end{aligned}$$

Therefore,

$$F_i^{\Omega_m} = F_i^{\Gamma_m}$$

Appendix B
Formulas for Residuals

Plane strain

$$R_y = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \cdot \frac{\partial^2 u}{\partial y^2} + \frac{E}{2(1+\nu)} \cdot \frac{\partial^2 u}{\partial z^2}$$

$$+ \frac{E}{2(1+\nu)(1-2\nu)} \cdot \frac{\partial^2 v}{\partial y \partial z} + f_y$$

$$R_z = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \cdot \frac{\partial^2 v}{\partial z^2} + \frac{E}{2(1+\nu)} \cdot \frac{\partial^2 v}{\partial y^2}$$

$$+ \frac{E}{2(1+\nu)(1-2\nu)} \cdot \frac{\partial^2 u}{\partial y \partial z} + f_z$$

Plane stress

$$R_y = \frac{E}{1-\nu^2} \cdot \frac{\partial^2 u}{\partial y^2} + \frac{E}{2(1+\nu)} \cdot \frac{\partial^2 u}{\partial z^2}$$

$$+ \frac{E}{2(1-\nu)} \cdot \frac{\partial^2 v}{\partial y \partial z} + f_y$$

$$R_z = \frac{E}{1-\nu^2} \cdot \frac{\partial^2 v}{\partial z^2} + \frac{E}{2(1+\nu)} \cdot \frac{\partial^2 v}{\partial y^2}$$

$$+ \frac{E}{2(1-\nu)} \cdot \frac{\partial^2 u}{\partial y \partial z} + f_z$$

Axisymmetric

$$R_y = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)}\cdot\frac{\partial^2 u}{\partial y^2} + \frac{E}{2(1+\nu)}\cdot\frac{\partial^2 u}{\partial z^2}$$

$$+ \frac{E}{2(1+\nu)(1-2\nu)}\cdot\frac{\partial^2 v}{\partial y\partial z}$$

$$+ \frac{E(1-\nu)}{(1+\nu)(1-2\nu)}\cdot\frac{1}{y}(\frac{\partial u}{\partial y} - \frac{u}{y}) + f_y$$

$$R_z = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)}\cdot\frac{\partial^2 v}{\partial z^2} + \frac{E}{2(1+\nu)}\cdot\frac{\partial^2 v}{\partial y^2}$$

$$+ \frac{E}{2(1+\nu)(1-2\nu)}\cdot\frac{\partial^2 u}{\partial y\partial z} + \frac{E}{2(1+\nu)(1-2\nu)}\cdot\frac{1\partial u}{y\partial z}$$

$$+ \frac{E}{2(1+\nu)}\cdot\frac{1}{y}\frac{\partial v}{\partial y} + f_z$$

Three-dimension

$$R_x = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)}\cdot\frac{\partial^2 u}{\partial x^2}$$

$$+ \frac{E\nu}{(1+\nu)(1-2\nu)}(\frac{\partial^2 v}{\partial x\partial y} + \frac{\partial^2 w}{\partial x\partial z})$$

$$+ \frac{E}{2(1+\nu)}(\frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} + \frac{\partial^2 v}{\partial x\partial y} + \frac{\partial^2 w}{\partial x\partial z}) + f_x$$

$$
\begin{aligned}
R_y \;=\; & \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \cdot \frac{\partial^2 v}{\partial y^2} \\
& + \frac{E\nu}{(1+\nu)(1-2\nu)}\left(\frac{\partial^2 u}{\partial x \partial y} + \frac{\partial^2 w}{\partial y \partial z}\right) \\
& + \frac{E}{2(1+\nu)}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial z^2} + \frac{\partial^2 w}{\partial y \partial z} + \frac{\partial^2 u}{\partial x \partial y}\right) + f_y
\end{aligned}
$$

$$
\begin{aligned}
R_z \;=\; & \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \cdot \frac{\partial^2 w}{\partial z^2} \\
& + \frac{E\nu}{(1+\nu)(1-2\nu)}\left(\frac{\partial^2 u}{\partial x \partial z} + \frac{\partial^2 v}{\partial y \partial z}\right) \\
& + \frac{E}{2(1+\nu)}\left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 u}{\partial x \partial z} + \frac{\partial^2 v}{\partial y \partial z}\right) + f_z
\end{aligned}
$$