SEMM REPORT
UCB/SEMM-90/09

STRUCTURAL ENGINEERING,
MECHANICS AND MATERIALS

AN UNSTRUCTURED QUADRILATERAL MESH GENERATOR: APPLICATION TO ADAPTIVE MESH REFINEMENT

by
S. Sakai and R.L. Taylor

JULY 1990

DEPARTMENT OF CIVIL ENGINEERING
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA

An Unstructured Quadrilateral Mesh Generator: Application to Adaptive Mesh Refinement

Shinsuke Sakai

Department of Marine Engineering, Faculty of Engineering

University of Tokyo

7-3-1 Bunkyo-ku Hongo, Tokyo 113, Japan

Robert L. Taylor

Structural Engineering, Mechanics and Materials

Department of Civil Engineering

University of California at Berkeley

Berkeley, CA94720

ABSTRACT

A new algorithm for an unstructured mesh generator which forms all quadrilateral elements is described. The quadrilateral elements are generated according to some measure, such as an error indicator evaluated by a finite element solution. The mesh generator is applied to exponentially distributed functions and is found to be applicable even for high gradient regions. Mesh adaptation is performed by regenerating using information obtained from the solution on the current mesh. The method is found to be reliable even for complicated boundary shapes.

Table of Contents

Acknowledgements by first author

This research has been performed during my stay at the Department of Civil Engineering at U.C. Berkeley as a Visiting Scholar. I have been supported by a grant from the Japanese Educational Ministry. During my research, my mother-in-law was found to be suffering from a serious illness and my wife had to return to Japan for a long period. My daughter Chie and I were given support in every aspect of daily life and in my research from the following persons:

Prof. R.L.Taylor (Host professor),
Panayiotis Papadopoulos,
the Thomas family,
the Takewaki family,
the Minami family,
the FitzGerald family,
the Hara family,
Prof. T.Hisada,
Priest Kubodera,
Mrs. Tanner,
Mrs. Lee,
Nigel Ward,
Kirk Martini,
my father-in-law,
and my parents

P.S.

I would like to acknowledge their hospitality and encouragement, I also hope that this research will contribute greatly to the progress of the field of the finite element method. *My mother-in-law died on July 3rd.*

1. Introduction

Development of an unstructured mesh generator is important for performing adaptive mesh refinement effectively. By using an unstructured mesh generator, it becomes possible not only to automatically handle problems with complicated boundaries, but also to handle rapid variations in mesh density. An unstructured triangular mesh generator already has been developed by Peraire et. al.[1] and has been applied to stress concentration problems in elastic solids and the shocks in compressible flow problems. In general, however, quadrilateral elements are used more frequently than triangular ones. Thus, the development of an unstructured quadrilateral mesh generator is greatly desired.

A quadtree algorithm intended for an unstructured mesh generator has been developed by Baehmann et. al.[2], but an all-quadrilateral mesh is rather difficult to achieve with this. Although the parallel mesh generation algorithm[3] is intended for all-quadrilateral elements, this method has difficulty in handling general boundary geometries precisely. Furthermore, it is difficult to apply these methods to high gradient region because the mesh density generated varies too greatly.

In this report, an unstructured quadrilateral mesh generation algorithm is proposed based upon the concept of the unstructured triangular mesh generation[1]. The method is applicable to arbitrary boundary geometries and is applicable even to high gradient regions. The method is applied to adaptive mesh refinement of some elastic analysis and satisfactory results are obtained.

2. Generation of All-Quadrilateral Elements

2.1 Definition of local parameters

It is shown by Peraire et.al.[1] that adaptive mesh refinement in high gradient regions can be performed effectively by using an unstructured triangular mesh. Figure 1(a) shows an application of their method to a supersonic flow problem. This method does not require complicated procedures (such as sub-division of the region) even if the domain boundary geometry is complicated. Therefore, an unstructured mesh generator has the advantage that adaptive mesh refinement can be performed fully automatically. Peraire et. al. define three local parameters to characterize the unstructured triangular mesh as shown in Fig.(b);δ:node spacing, s:stretching parameter, and α:direction of stretching. The stretching parameter plays an important role especially in high gradient regions because it predicts whether the trend of the side distribution toward the direction of generation is increasing or decreasing.

For any unstructured quadrilateral mesh generator, these three local parameters are necessary and are introduced accordingly as h,s and α in Fig.2. The local parameters have to be defined at every point in the solution domain. When it is applied to a FEM, the node spacing h at the point of interest is evaluated by interpolation over a background mesh. Therefore, the h values have to be specified at each node a priori.

For the initial mesh, the solution domain will be expected to be filled with a single element if the following conditions are satisfied.

- h = constant
- s = 1
- α: the direction normal to the bottom side.

Unstructured triangular mesh generator

by J. Peraire et. al.

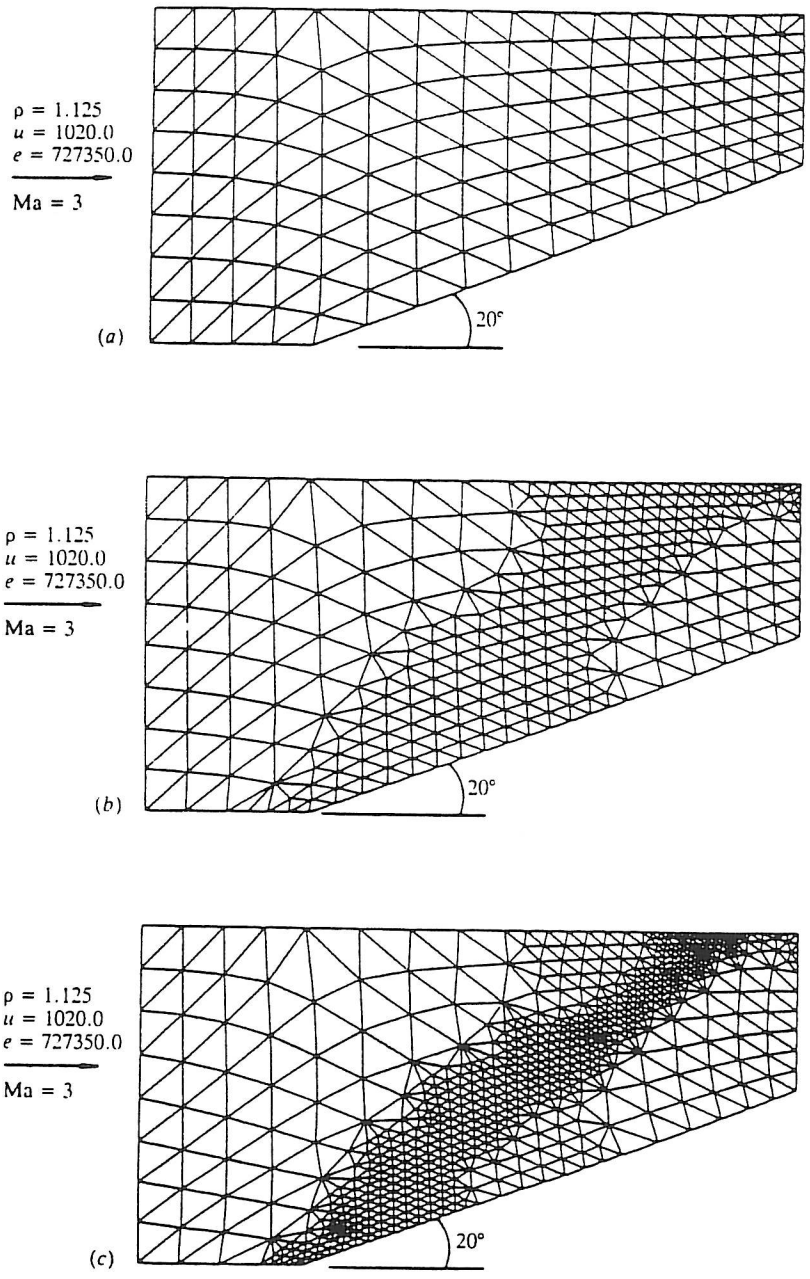


Fig. 14.19 Mesh enrichment. Supersonic flow past wedge (Mach number 3). (*a*) Initial configuration of mesh. (*b*) After 101 steps. (*c*) After 201 steps (ρ = density; u = velocity; e = specific energy)

(a)

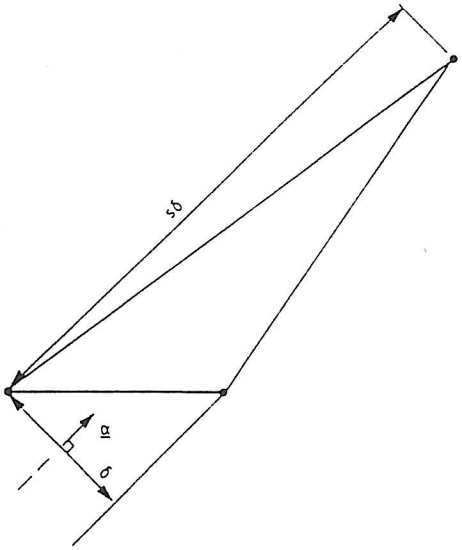


FIG. 2.2. The definition of the mesh parameters δ, s, and α.

(b)

Fig.1 An example of unstructured triangular mesh generation[1)]

h projected node spacing

α direction of steepest gradient

S stretching parameter = $dh/d\alpha$

——— active front mesh

------- background mesh

— — — generated mesh

× location to evaluate local parameter

○ node

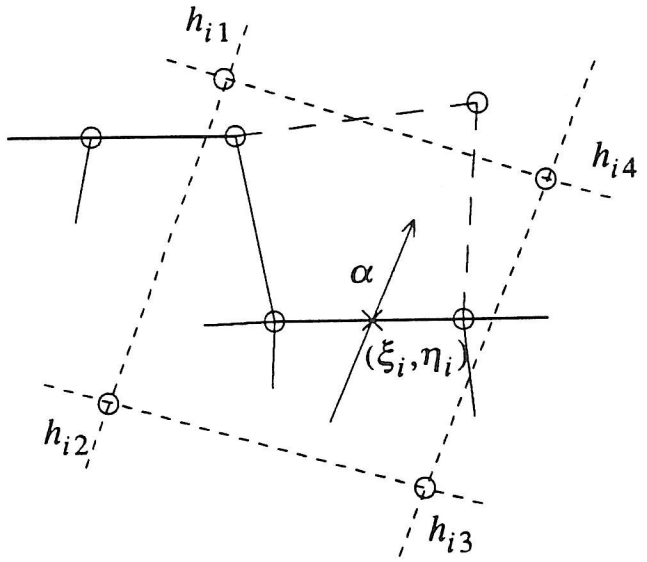


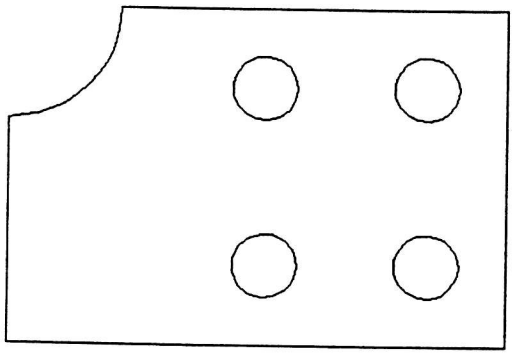
Fig.2 Definition of local parameters

2.2 Introduction to generation procedure

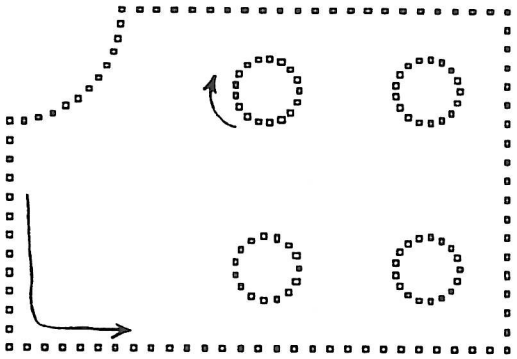
Figure 3 shows the process adopted for an unstructured quadrilateral mesh generation. This section sketches the procedure and details are discussed in subsequent sections. Generally speaking, the boundary geometry for a two dimensional problem can be described by the union of one exterior boundary loop and several interior boundary loops. Usually, mesh generation becomes troublesome for complicated boundary geometry. If the concept of an unstructured mesh generation is utilized, however, the procedure is significantly simplified even for the complicated boundary case. Thus, the generation procedure is performed exactly in the same manner independent of the boundary geometry. The proposed method performs adaptive mesh generation using the boundary geometry together with a distribution function defined in the solution domain. When it is applied to a FEM, the distribution function is estimated from an error indicator and is used as a measure to update the length of the element sides in regenerating the mesh.

The procedure begins with placing nodes on the boundary segments based upon the current distribution function $h(x,y)$(Fig.(a)). The boundary nodes are generated counter-clockwise for the exterior boundary and clockwise for the interior boundaries. In Fig.(b), each boundary line is considered in turn and elements are generated according to $h(x,y)$. In generating the mesh, the concept of a generation front, as proposed by Lo[3], is utilized. This concept is explained in the section 2.4. The mesh will be generated counter-clockwise for the exterior boundary and clockwise for the interior boundary in the same way as the boundary nodes. This means that, as the boundary curve is traversed, the region to be generated always lies to the right. As the generation proceeds, the boundary segments will collide with one another and will be combined together as shown in Fig.(c). This will produce an increase or decrease in the number of boundary segments. In this way, the remaining region will be filled with mesh gradually. However, to form all-quadrilateral elements from the beginning is quite difficult because overly distorted elements are liable to be produced especially in high gradient regions. In addition, it will be impossible to produce only quadrilaterals if the number of the remaining nodes on an active front is odd. Thus, though the present method generates quadrilaterals as far as possible, it will form triangular elements whenever the quadrilateral formed is evaluated to

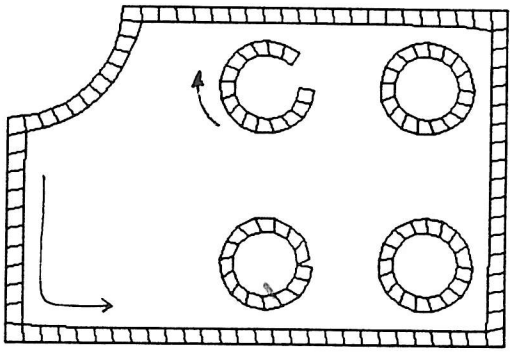
be too distorted or forming a quadrilateral is no longer possible. This means that a mixed mode mesh of 3-node elements and 4-node elements is produced at this stage. Figure (d) shows the state after generating the mixed mode mesh. To achieve all-quadrilateral elements, the triangular elements have to be merged into quadrilateral ones in the next step. This procedure is basically performed by changing the connectivities of elements;Fig.(e) shows the state after the process. At this stage, there are many distorted quadrilateral elements in the domain and a smoothing process is required. This leads to the last figure (f).



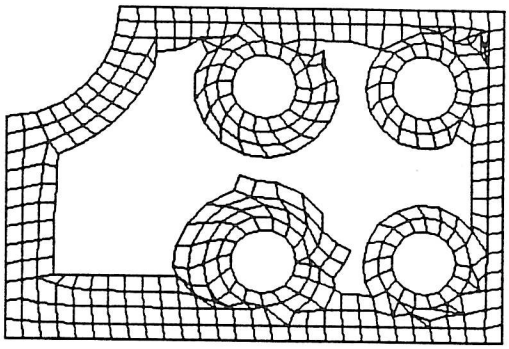
Solution domain



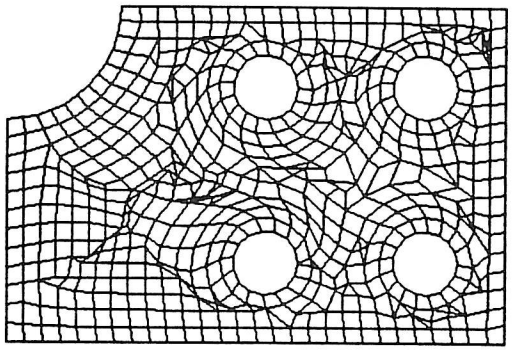
(a)



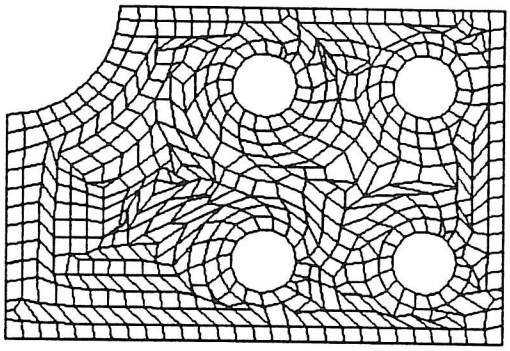
(b)



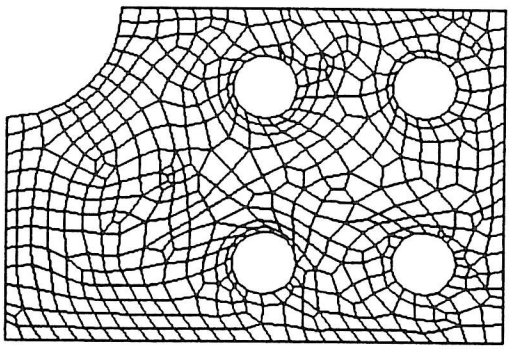
(c)



(d)



(e)



(f)

Fig.3 Procedure for mesh generation

2.3 Generation of boundary nodes

Boundary nodes are placed according to the distribution function $h(x,y)$. Figure 4 shows the method for placing the boundary nodes. Suppose the local parameters at the previously generated node n_i are $h_i, s_i and, \alpha_i$. α has to be taken in the direction tangent to the boundary line. To generate the next boundary node n_{i+1}, the node a which is at distance $(1+s_i)h_i$ from n_i has to be determined first. This point a is not adequate as n_{i+1} because the edge length may change rapidly by this method. To avoid this, an average edge length between edge value at n_i and one at a is calculated as

$$H = \frac{1}{2}\left\{(1 + s_i)h_i + (1 + s_a)h_a\right\}$$

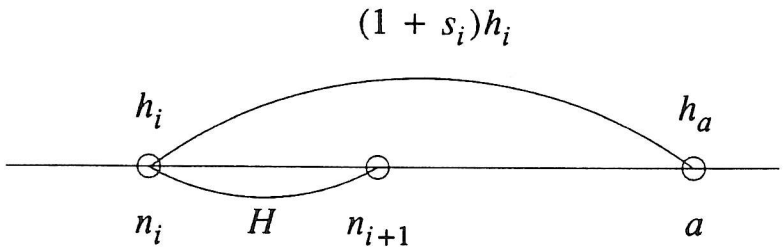
where, s_a, h_a are the local parameters at point a. Then, the point which is at distance H from n_i is allocated as the new node n_{i+1}.

When the generation of boundary nodes reaches the corner node m as shown in Fig.(b), the treatment is as follows.

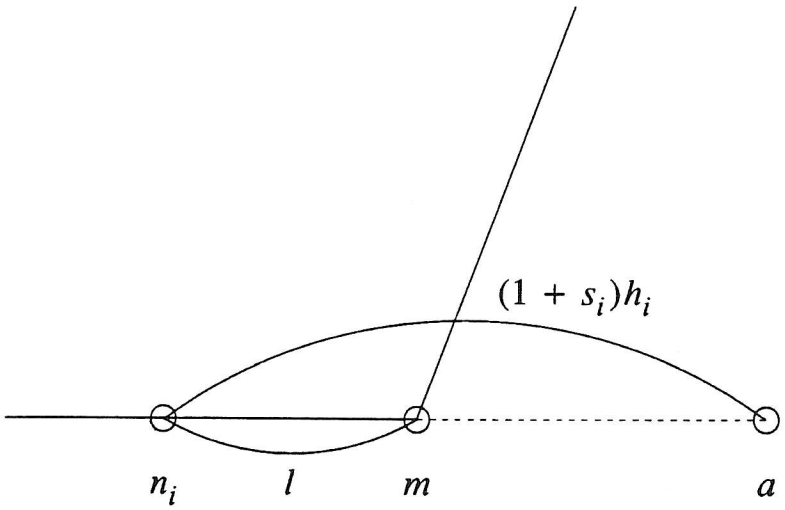
First, calculate a ratio r from

$$r = \frac{l}{(1 + s_i)h_i}$$

where, l is the distance between m and n_i. If $r \geq 0.5$, m is allocated as n_{i+1}. If not, the previously generated node n_i is removed and the node m is allocated as n_i.



(a)

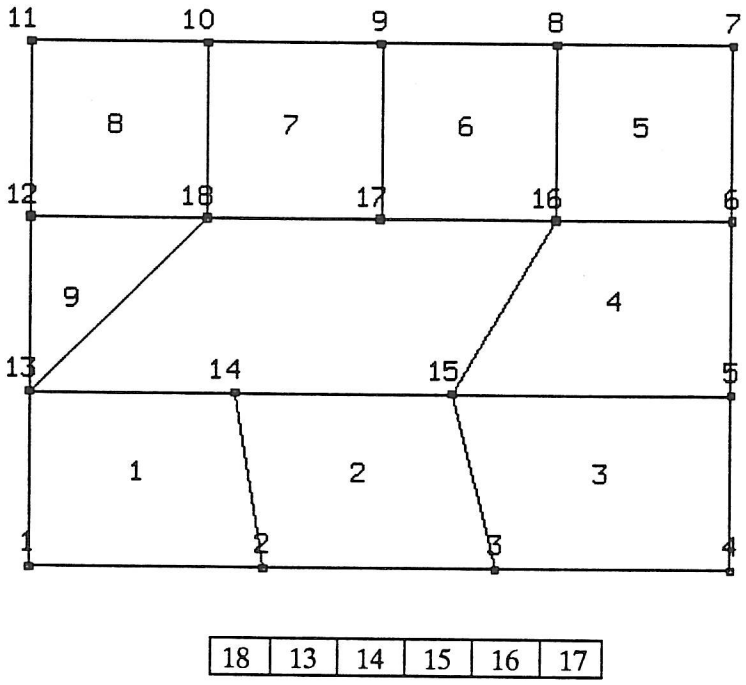


(b)

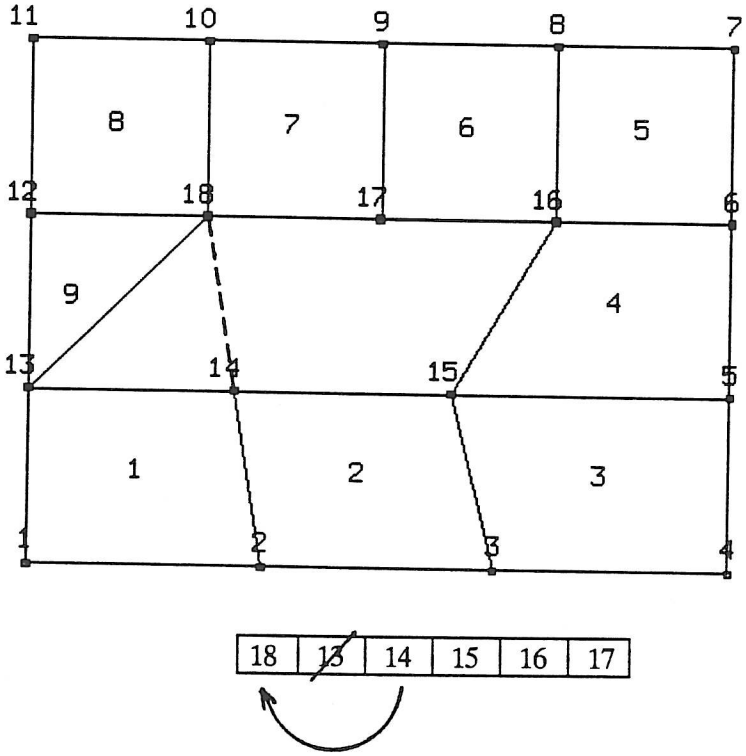
Fig.4 Procedure for generating a boundary node

2.4 Concept of generation front

The generation algorithm uses the concept of a generation front as proposed by Lo[4]. Although this concept has been applied only to triangular elements to date, it will be used for quadrilateral elements in this report. During the generation process, consecutive straight lines which are available to generate a new element are termed active. At the start of the process, the consecutive lines which connect the boundary nodes are the active front. Once a new element is generated, nodes which are no longer active are removed from the front and the active front is updated. Thus the active front changes continuously as the generation process proceeds. Figure 5 shows an example of the process for updating the front table. At a certain stage the generated mesh and the active front table are as given in Fig.(a). When element 10 is newly generated as shown by the broken line in Fig.(b), node 13 is no longer active and is removed from the current table. The table then is linked as shown below the figure.



(a) The current front at a certain stage



(b) The updated front after generation of a new element

Fig.5 Updating process for the front table

2.5 Generation of mixed mode mesh

Figure 6 shows a method to form a new element from the front nodes. The steps to generate a new element are as follows:

(a) Suppose the first three nodes in the current active front are n_1, n_2, n_3 as shown in Fig.(a). Evaluate the angle $\theta = n_1 n_2 n_3$. The generation procedure differs depending on whether θ is greater than θ_0 or not. There is no unique choice for the value of θ_0 which should be adopted, but $\theta_0 = 0.75\pi$ is used for the computations shown in this report.

(b) Determine the local mesh parameters h, s and α at the midpoint M of $n_2 n_3$ by interpolating over the background grid.

For $\theta < \theta_0$:

(c) Determine two points: one at a distance $(1+s)h$ from node n_1 and normal to α, the other at a distance $(1+s)h$ from node n_3 and parallel to α. A candidate node is generated at the mid-point of these two points.

(d) Determine all the active nodes which lie within a circle with center at the candidate node and radius $r|n_2 n_3|$. In this report, $r = 5$ has been used for the computations. These nodes are ordered according to their distance from the candidate node and are denoted by $M_1, M_2, \ldots, M_k$. Thus, M_1 is the closest point to the candidate node in the list.

(e) Place the candidate node at the head of the list.

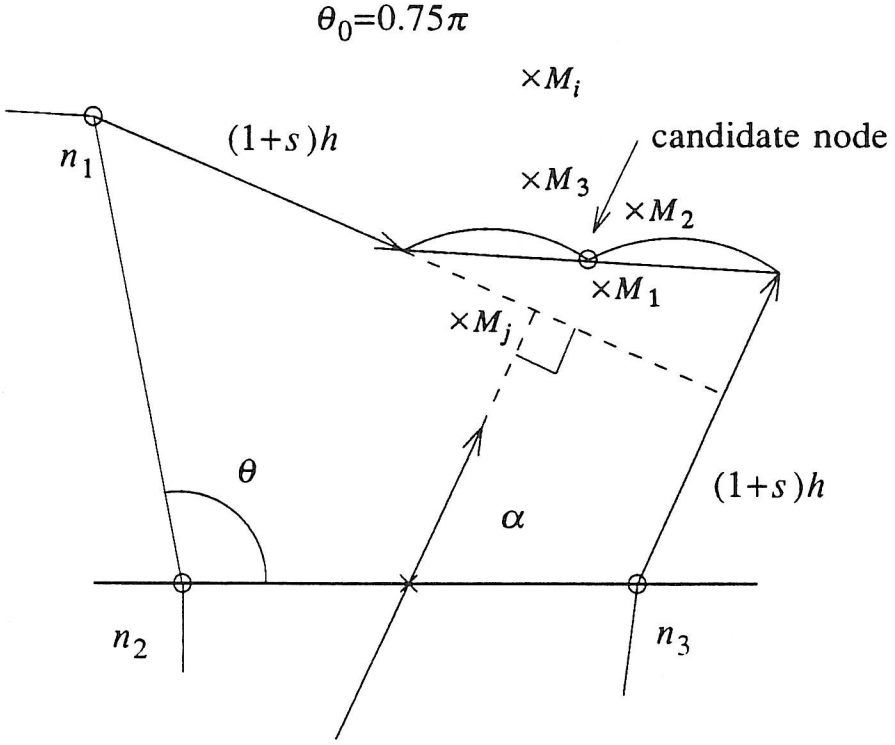
(f) The new node M_j is chosen as the first point in the list which is such that the interior of the quadrilateral $n_1 n_2 n_3 M_j$ does not contain any other point M_i in the list and such that the line $n_1 M_j$ and the line $n_3 M_j$ do not intersect any edges in the current front. The new element $n_1 n_2 n_3 M_j$ is generated and the front table is updated as described in the previous section(e.g., see Fig.5). If the number of active edges in the front table becomes zero, the generation process ceases. Go to (a), otherwise.

For $\theta \geq \theta_0$:

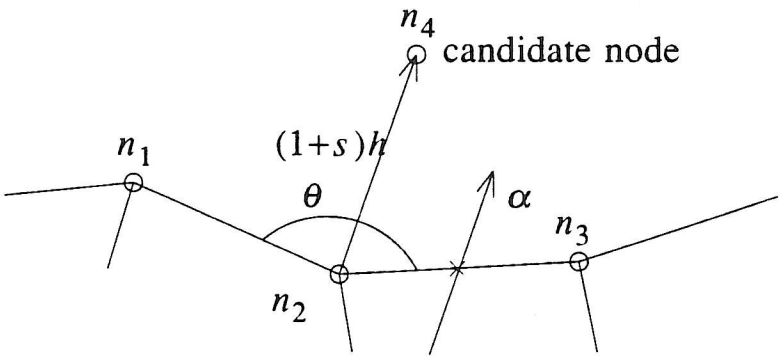
(c) A candidate node n_4 is determined at the point which is at a distance $(1+s)h$ from node n_2

and is parallel to α.

(d) Update the front table as indicated in Fig.5 and place n_4,n_2,n_3 as the first three nodes in the table. Go to (a).



(a) for $\theta<\theta_o$



(b) for $\theta\geq\theta_o$

Fig.6 Procedure for generating a new node

2.6 Searching algorithm

Figure 7 shows the principle of the algorithm. The values of the parameters h ,s and α are needed to construct the mixed mode mesh. To accomplish this, a search over the background elements is made to locate the element which contains the midpoint of n_2n_3. The searching algorithm is implemented efficiently as follows.(see Figure 7.):

(a) for the background mesh build a table which identifies the list of elements connected to each element as shown in Fig.7.

(b) Given a global coordinates (x,y) for a point M, the natural coordinates (ξ,η) are determined from the isoparametric mapping function for element e.

(c) If $-1 \leq \xi \leq 1$ and $-1 \leq \eta \leq 1$, M is contained within the element; else.

(d) consider the 4 surrounding elements which have sides in common with element e and identify the number of the region which contains M.(e.g. region number 3 in Fig.7)

(e) If the region number is i, e_i is the next element to be checked.(e.g. element e_3 in Fig. 7) Go to (b).

In this way, searching all elements is avoided and the searching time to find the appropriate background element containing M will be greatly reduced.

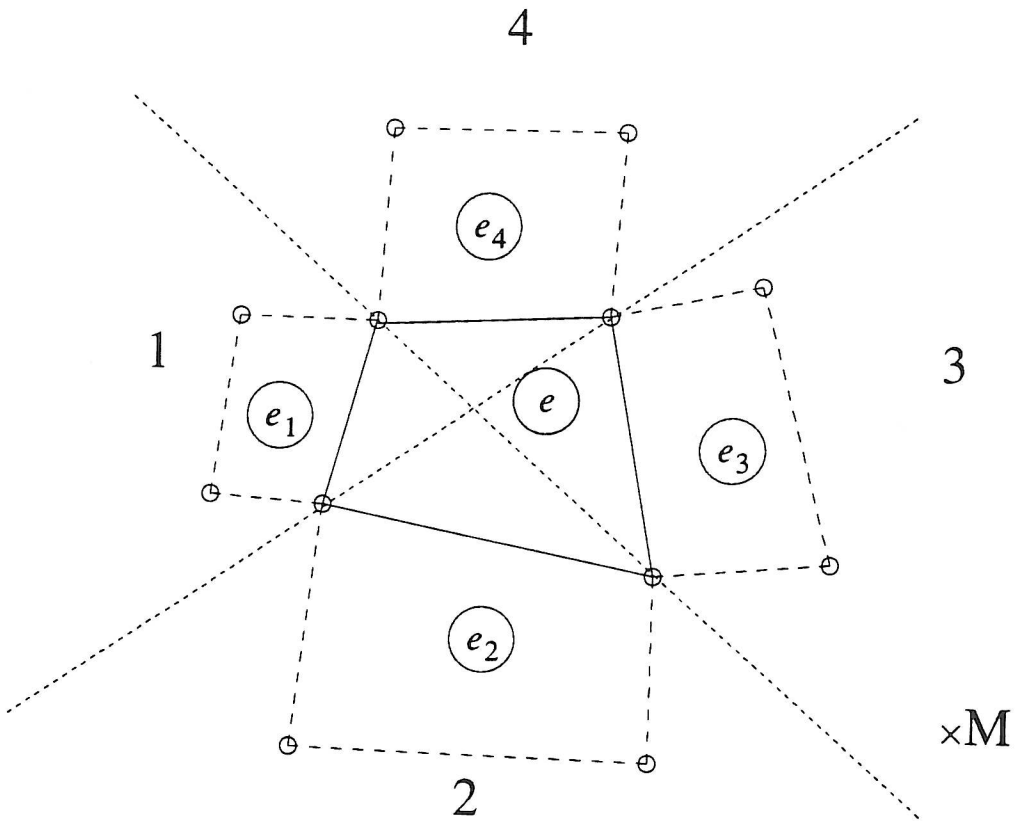
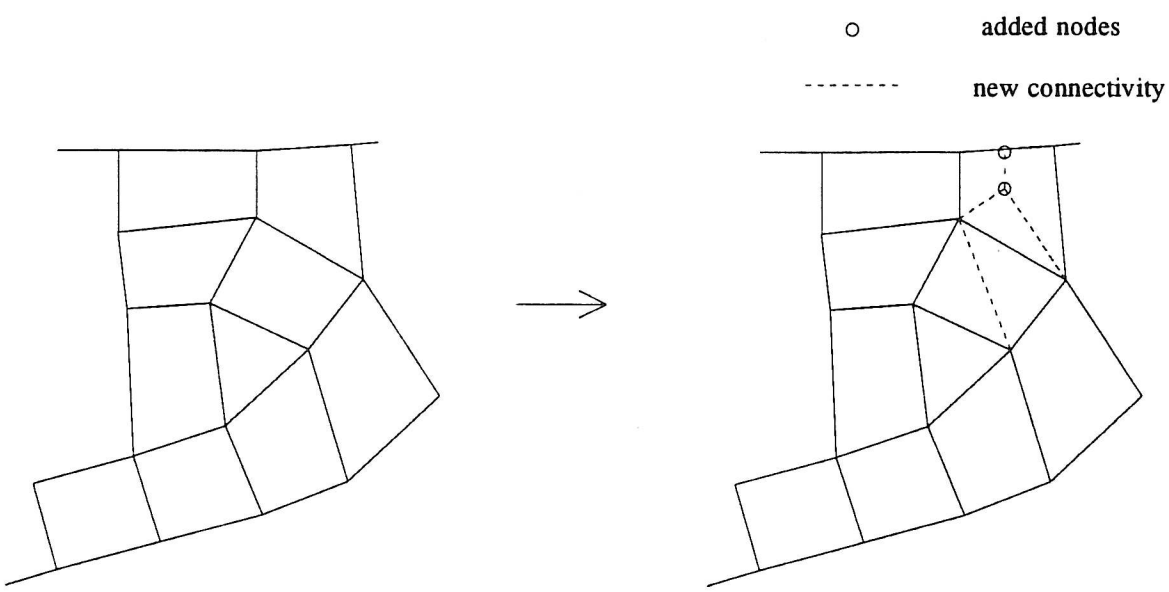


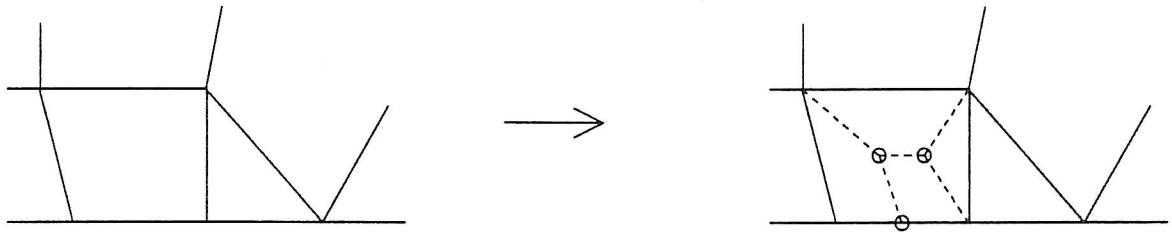
Fig.7 Searching algorithm

2.7 Merging of 3-node elements

After generating a mixed mode mesh, triangular elements must be eliminated to achieve an all-quadrilateral mesh. Figure 8 shows a simple method to achieve this step. The merging step begins with the triangular with highest element number. If a triangular element exists in an interior region, it is merged into a quadrilateral one by changing the connectivities of elements towards the boundary line as shown in Fig.8(b). Whenever a triangular element collides with a pre-existing triangular element during the process, a quadrilateral element is formed from these two elements. Since the mixed mode generation has been performed from the exterior to the interior, the direction to the boundary line will be identified easily because elements nearer their boundary have lower element numbers. In forming a new quadrilateral element the diagonal of the adjacent quadrilateral which yields the least distorted shape is selected. When the merging procedure reaches the boundary line, two new nodes are generated within the boundary element to produce quadrilaterals as shown by the broken lines in Fig.8(b). It may be rather troublesome to generate a new node at the midpoint of the two boundary nodes, but no more than one midpoint in the boundary side is required in this process. For the boundary triangular element, three new nodes are generated within the adjacent boundary quadrilateral element and the connectivities of the elements are changed as shown by broken lines in Fig.(a).



(b)In the case of an interior triangular element



(a)In the case of a boundary triangular element

Fig.8 Merging 3-node elements into 4-node elements

2.8 Modified Laplacian operator

There are several higher distorted quadrilateral elements at the end of the merging process(Fig.3 (e)). To improve their shapes, a smoothing process is performed as the final step. A Laplacian smoothing operator[5] has been widely used as a smoothing method.(e.g. see Fig. 9) In the Laplacian smoothing method, the interior nodes in the object are repositioned to improve the shape of the quadrilaterals. The repositioning of the interior nodes is accomplished using a weighted form of the Laplacian smoothing method. In this method, an interior node is positioned at a weighted centroid of the nodes that define the quadrilaterals surrounding that node. In calculating the centroid, a weight of 2 is assigned for nodes associated with edges and a weight of 1 is used for nodes on diagonal of each element. This weighted form Laplacian smoothing method can be expressed as

$$T = (\sum_{i=1}^{N} P_i + 2\sum_{j=1}^{M} Q_j)/(N + 2M)$$

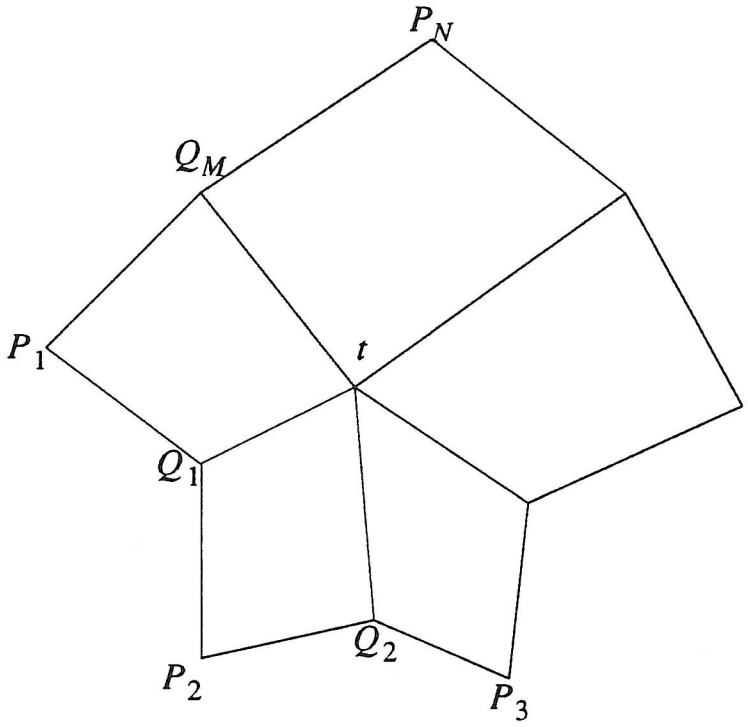
where T is the new location of the interior node. P_i are the coordinates of node i, which is on a diagonal of the quadrilateral, Q_j are the coordinates of node j, which is on an edge containing the node being smoothed. N is the number of nodes for P_i and M is the number of nodes for Q_j. These are shown in Fig.(a). From experience, however, this simple method often produces distorted elements such as shown in Fig.(b). One of the quadrilaterals is nearly a triangular and a near zero Jacobian may occur during finite element analysis due to the distorted element. To avoid this, the values for the weighted form of the Laplacian smoothing method are slightly modified. For the nodes on element diagonal, the weighted values are modified according to the corresponding open angles at the interior node as shown in Fig. 10. The modified Laplacian operation can be expressed as

$$T = (\sum_{i=1}^{N} \alpha_i P_i + 2\sum_{j=1}^{M} Q_j)/(\sum_{i=1}^{N} \alpha_i + 2M)$$

where the meanings of P_i, Q_j, N, M are the same as those described above. For α_i, the usual Laplacian smoothing method always uses 1, but the present method utilizes the values shown in Table 1. There is no unique choice for the values of $\alpha_i, \theta_{max}, \theta_{min}$, however based upon experience in solving several problems, the values in Table 1 give improved results. In the present operator, the interior node is pulled in the direction of the opposite node if $\theta > \theta_{max}$ and is pushed away from the opposite

node if $\theta \leq \theta_{max}$.

Laplacian smoothing method



(a)

$$T = (\sum_{i=1}^{N} P_i + 2\sum_{j=1}^{M} Q_j)/(N + 2M)$$

T:new location of node t

this may produce

(b)

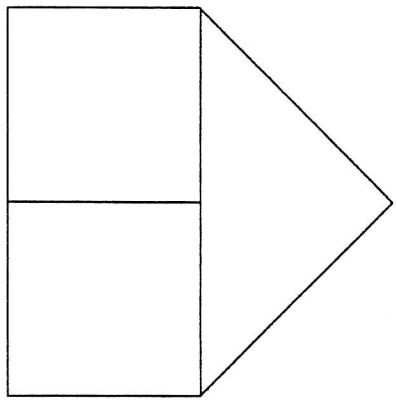
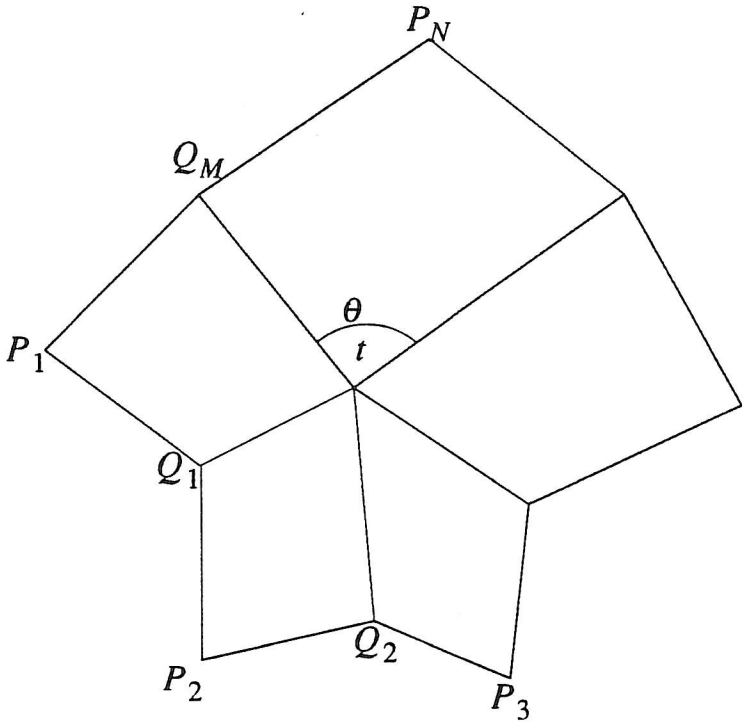


Fig.9 The Laplacian smoothing method

Modified Laplacian smoothing method



$$T = (\sum_{i=1}^{N} \alpha_i P_i + 2\sum_{j=1}^{M} Q_j)/(\sum_{i=1}^{N} \alpha_i + 2M)$$

Fig.10 The modified Laplacian smoothing method

Table 1 Modified weighted values

		Modified	*Standard*
$\alpha =$	$\theta > \theta_{max}$	0.0	1.0
	$\theta \leq \theta_{min}$	2.0	1.0
	otherwise	1.0	1.0

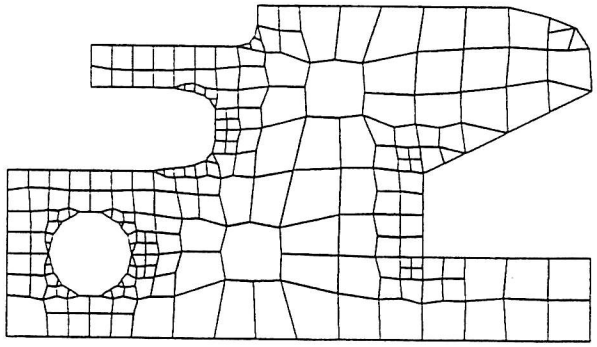
$$\theta_{max} = 0.8\pi$$
$$\theta_{min} = 0.2\pi$$

2.9 Examples

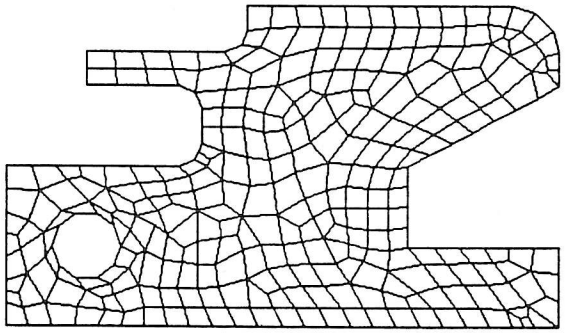
2.9.1 Complicated boundary geometry

The proposed method is applied to several domains with complicated boundary geometry and uniformly distributed mesh generations are performed. The unstructured mesh generator has the advantage that there is no limitation on the boundary geometry. Figure 11 shows the comparison of the mesh generated by the present method with one by the quadtree method of Baehmann et. al.[2] It is clear that the quadtree solution does not fill the domain with quadrilateral elements only. On the other hand, the present method achieves the all-quadrilateral elements as shown in Fig.(b). In Fig.(a), some mesh refinements are performed around the corners, but it seems that the edge length distribution does not change smoothly around each corner. For effective adaptive mesh refinement, it is necessary to produce a mesh varying smoothly according to some distribution function. It will be shown in 3.4.2 that the present method yields this performance.

Figure 12 shows the comparison of a mesh generated by the present method with one determined by the parallel mesh generation algorithm of Cheng et. al.[3]. The parallel mesh generation method has the advantage that it can be computed in parallel and naturally ensures the conformity of the generated mesh. In this method, an all-quadrilateral mesh is indeed achieved, as shown in Fig.(a). However, it can not represent the given boundary geometry precisely because its boundary line is determined at the first stage of mesh generation. In addition the rate of changing the mesh size is rather rapid. On the other hand, the present method, as shown in Fig.(b) can represent the given boundary line precisely and can achieve an all- quadrilateral mesh.

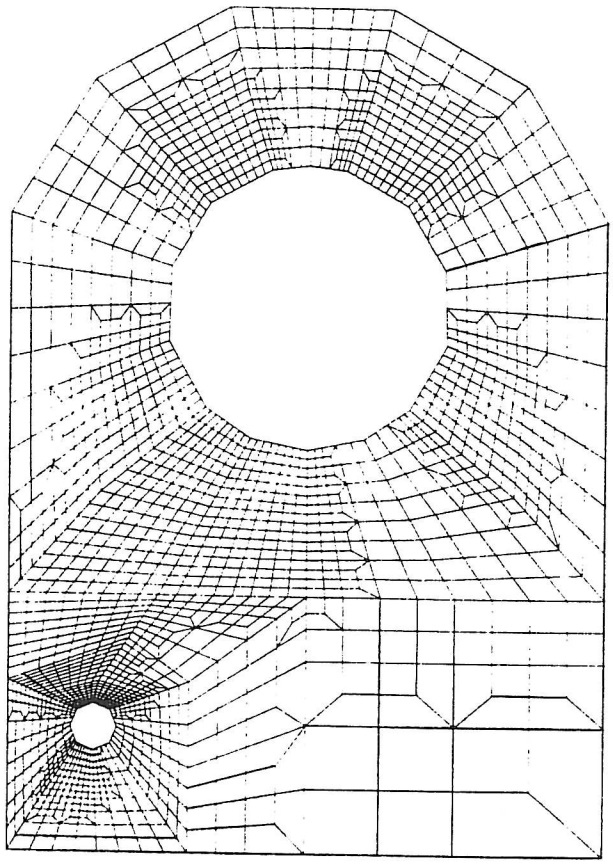


(a) Quadtree mesh generator by Baehmann et. al.

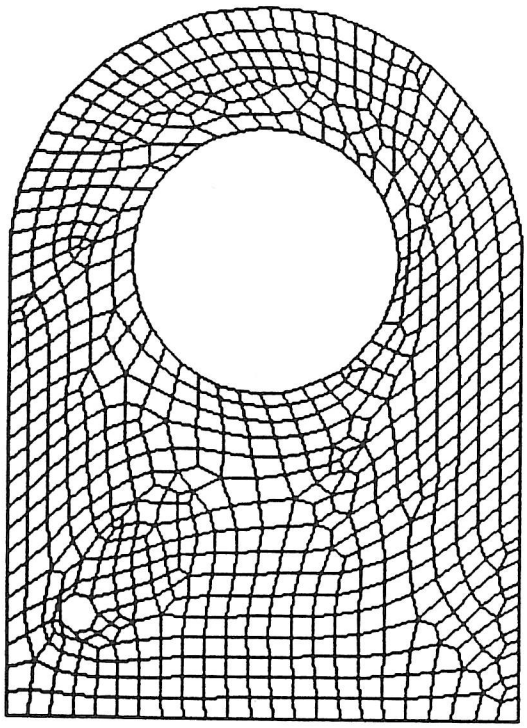


(b) Present algorithm

Fig.11 Comparison of the present algorithm with quadtree mesh generation one when applied to a complicated boundary domain



(a) Parallel mesh generation algorithm by Cheng et. al.



(b) Present algorithm

Fig.12 Comparison of the present algorithm with parallel mesh generation one when applied to a complicated boundary domain

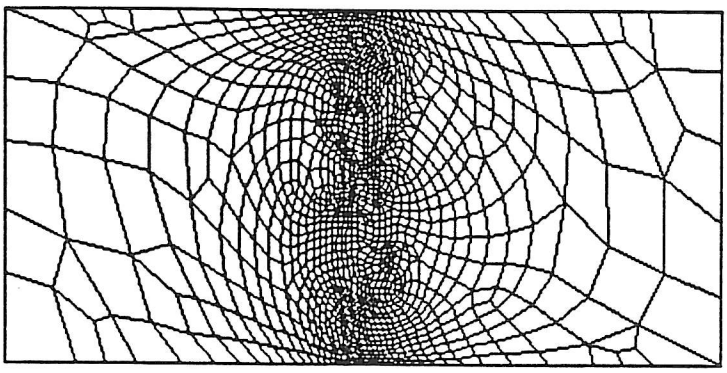
2.9.2 Generation for exponential distribution

The present method is able to generate a mesh based upon any arbitrary distribution function. To examine the adaptability of the present method to a given distribution function, a mesh generation is performed using an exponential function as the given distribution function. If the exponential function within the rectangular region is given as

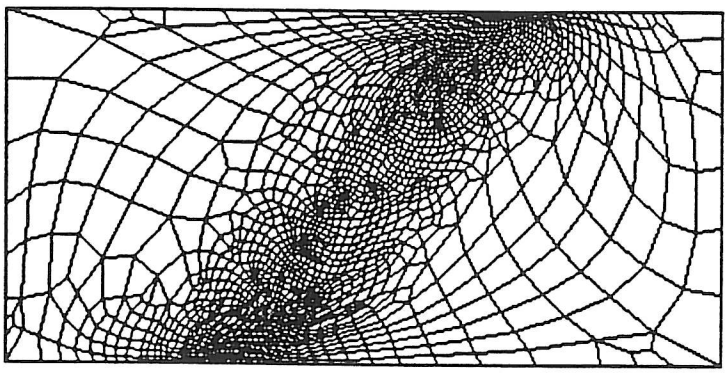
$$y = -Ae^{(-Bx^2)} + C$$

where A ,B ,C are constant parameters and are determined from the edge lengths of the elements at the rightmost side and at the center and the changing ratio of the side length. Quadrilateral elements are generated within a region with width l_x and height l_y according to the distribution function. Figure 13 shows the result for $A=1.011, B=0.106, C=1.071, l_x=10, l_y=5$. Figure (a) shows that the edge length does in fact becomes shortest on the y-axis. Even if the principal axis of the distribution function is inclined 40^o from y-axis, the mesh can be generated precisely according to the function as shown in Fig.(b).

Figure 14 shows the result when applied to an even higher gradient case. The values used are $A=0.648, B=0.1, C=0.678, l_x=10, l_y=5$. Even for this high gradient region, the adaptive mesh generation can be performed as shown in Fig.(a). Figure (b) shows an enlargement of the region enclosed by bold line in Fig.(a); and Fig.(c) shows a further enlargement of the region enclosed by the bold line in Fig.(b). This confirms that an all quadrilateral mesh can be achieved even in a high gradient region. Thus, the proposed method is expected to be applicable to the vicinity of a singular point in stress analysis or shocks in a supersonic flow problem.

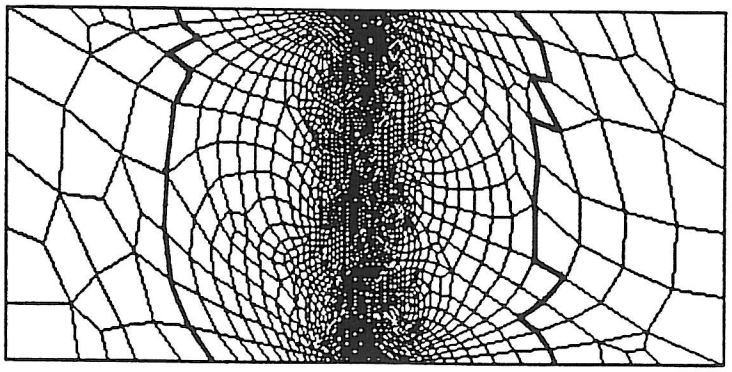


(a) Principal axis is in the normal direction

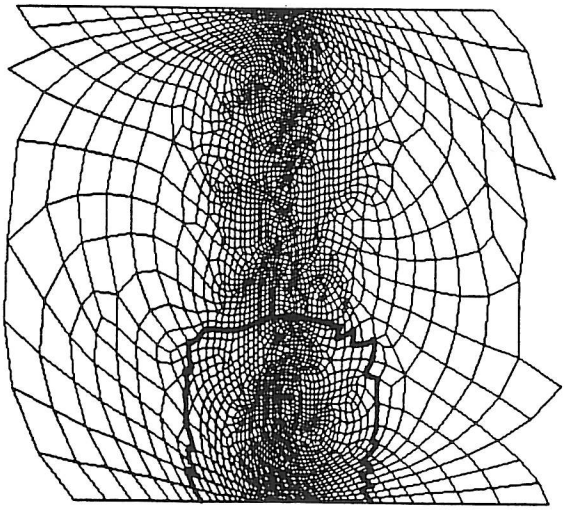


(b) Principal axis is rotated 40°
from the normal direction

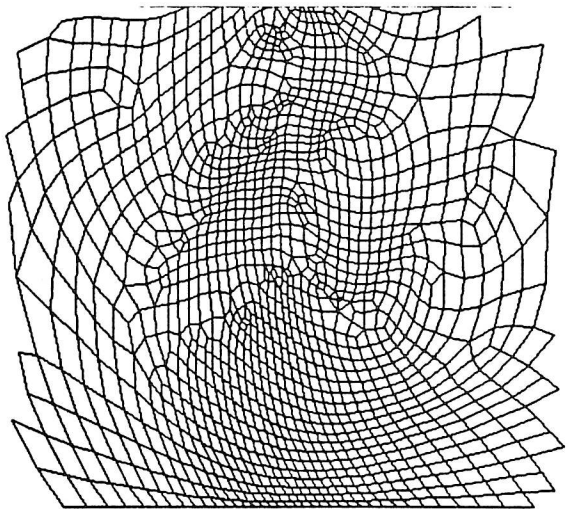
Fig.13 Generated mesh for the node spacing distribution according to an exponential function



(a) Whole domain



(b) Enlargement of the region enclosed by the bold line in fig.(a)



(c) Enlargement of the region enclosed by the bold line in fig.(b)

Fig.14 Application to the high gradient distribution function

3. Application to Adaptive Mesh Refinement

3.1 Procedure

In this section, the application of the proposed generation method to an adaptive mesh refinement is presented. Figure 15 shows the flowchart of the mesh refinement cycle ,where the distribution function of node spacing is denoted by $h(x,y)$. Since there is no information about $h(x,y)$ at the beginning, we use a uniform distributed function for $h(x,y)$ at the beginning so that the procedure is general. In steps (a) to (d), the mesh generation is performed according to the algorithm given in the previous chapter. The mesh obtained at this stage has an extremely large band-width and profile and is not adequate for an implicit FEM solution. Therefore an optimization of the band and the profile is performed using "Profile-Front Minimization" or "PFM" algorithm proposed by Hoit and Wilson[6]. The **PFM** method requires little computer memory and can execute rapidly. After the optimization process, the conditions of analysis are added to the mesh data and a FE analysis is performed using **feap**[9]. Based upon the calculation result, we can evaluate an error indicator in each element, as defined in the next section. The analyst determines whether or not the error value is satisfactory. Generally speaking, an error level of 5% in energy is sufficient for many practical purposes. If the result meets the accuracy requirement, the procedure terminates. If not, $h(x,y)$ is updated based upon the error values. As the error value in each element determines the amount by which to modify the current edge length, the distribution $h(x,y)$ can be updated based upon the results of each analysis. The procedure then returns to step (a) and the mesh refinement is performed again. This cycle is repeated until the analyst is satisfied with the result.

Figure 16 shows the relationship between the programs and the files in the refinement cycle described above. A *G –file* contains information about the boundary geometry and an *I –file* contains the input data for the program **feap**. Each *E –file* contains information related to the error indicators. At the beginning we need only to prepare the *G –file*. The steps from (a) to (d) in Fig.15 are combined into a single program, **unif**, for the 1st cycle and are combined into a program, **adaptive**, for subsequent cycles. The program **unif** generates the mesh under the conditions that the distribution function of node spacing is uniform, the stretching parameter always is 1 and the direction

of α is normal to the side of the element. Executing **unif** and **pfm** will produce an $I-file^{(1)}$, and an $E-file^{(1)}$ will be obtained by using the $I-file^{(1)}$ as input to **feap**. The 2nd refinement is performed by applying these two files together with $G-file$ to the program **adaptive**. Then, executing **adaptive** and **pfm** will produce $I-file^{(2)}$, and $E-file^{(2)}$ will be obtained by supplying the $I-file^{(2)}$ to **feap**. This cycle continues until the desired convergence is achieved - generally within 3 or 4 steps.

$h(x,y)$: distribution function of node spacing

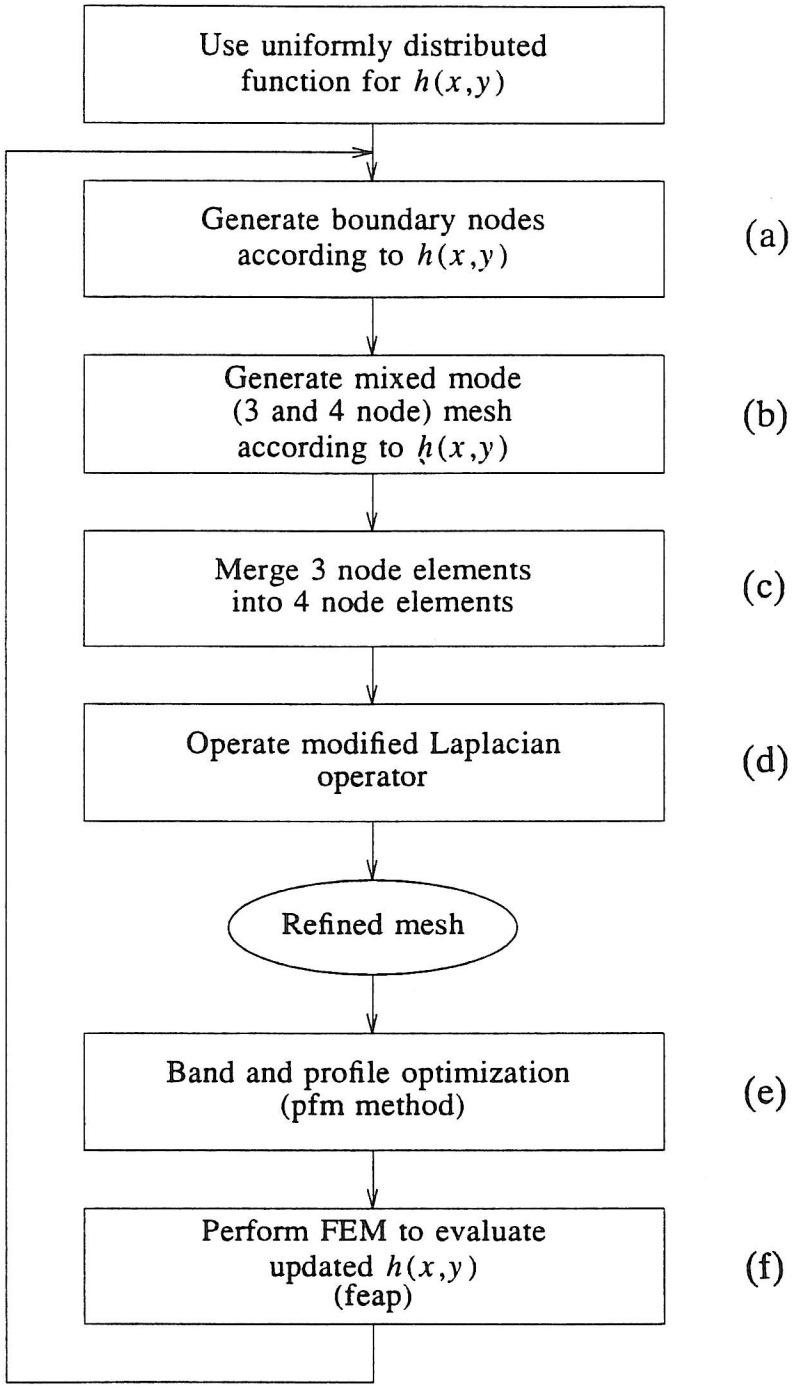


Fig.15 Mesh refinement cycle

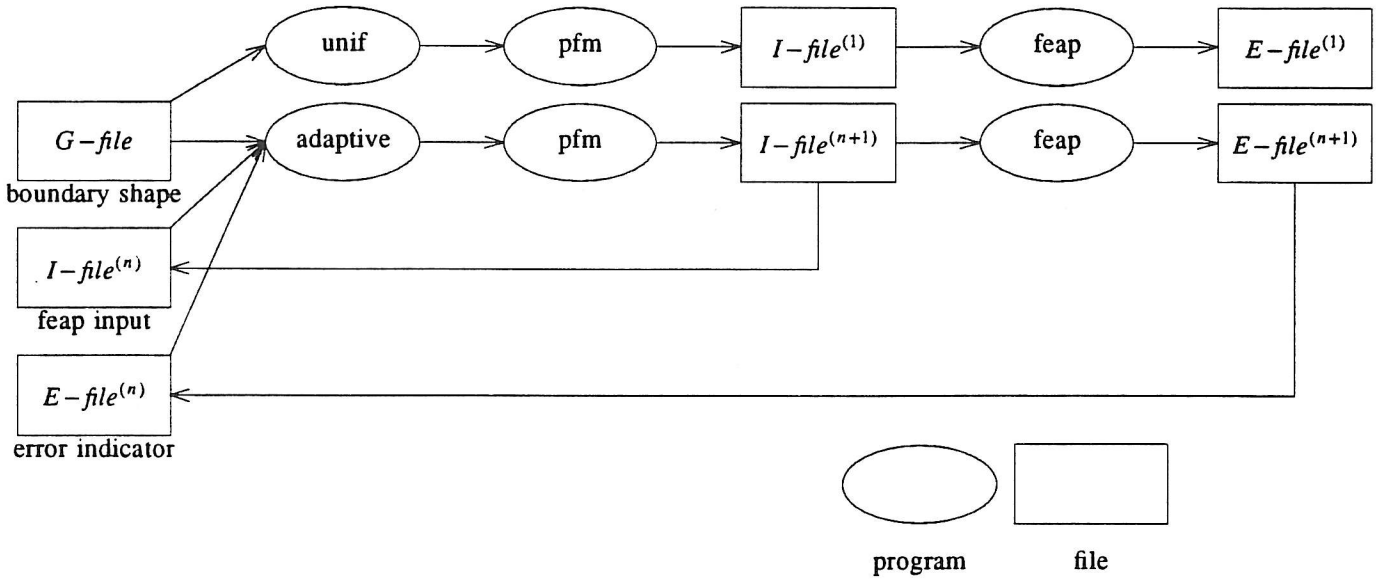


Fig.16 Schematic drawing of the adaptive mesh refinement cycle

3.2 Error estimation

In order to update the current edge length, an error indicator is used. As the exact solution is not known a priori, we define the error as follows. Based upon the calculated results of a FE analysis, a pseudo solution which is expected to be closer to the correct solution is evaluated first. The error is defined as the value obtained by integrating the difference between the pseudo solution and the calculated value over the element. It is well known[7] that the true error becomes bigger as the stress difference between two adjacent elements becomes bigger and the error definition adopted here is based upon this concept.

As for the pseudo solution, the nodal stress values $\hat{\sigma}_{ij}$ which are obtained by projecting the calculated stress values $\overline{\sigma_{ij}}$ to each node are used. Though there have been many stress projection methods, the node averaging method is used here because the principle is simple and it does not require much memory space. It is known that better results could be obtained by using the projection method proposed by Hinton and Campbell[8]. This method, however, would require us to solve a large linear equation again and so would require much memory space. The difference of the results between the two methods is rather small except when the mesh density is extremely coarse, and thus the node averaging method should be adequate for most cases. There are also many methods for defining errors from $\overline{\sigma_{ij}}$ and $\hat{\sigma}_{ij}$. The stress norm defined below is used for the computation in this report.

$$(verror)_i = \int_{\Omega_i} \sum_{ij} (\hat{\sigma}_{ij} - \overline{\sigma_{ij}})^2 d\Omega \tag{3.1}$$

where $(verror)_i$ is the error evaluated at element i and Ω_i is the area of the element. It is known that the energy norm will give better results and the authors will leave this as a topic for future research.

To update the edge length, calculate the average permissible error level e_m defined as

$$e_m = \bar{v}\left(\frac{eproj}{N}\right)^{1/2} \tag{3.2}$$

where $eproj$ is a sum of the integration of $\hat{\sigma}_{ij}$ which is defined as

$$eproj = \sum_e \int_{\Omega_e} \left(\sum_{ij} \hat{\sigma}_{ij}^2\right) d\Omega, \tag{3.3}$$

where, N is a number of elements and $\bar{\nu}$ is a permissible error level per each element. Usually, 5% is sufficient for the value of $\bar{\nu}$ for practical purposes. The error ratio ξ_i is defined as the ratio of the $(verror)_i$ to e_m:

$$\xi_i = \frac{(verror)_i}{e_m} \tag{3.4}$$

It is not necessary to update the edge length of an element if $\xi_i = 1$. The edge length of the elements where ξ_i is greater must be made smaller. Thus, the current edge length h_i is updated according to the following equation:

$$h_i' = h_i \left(\frac{1}{\xi_i}\right)^{-1/p} \tag{3.5}$$

where p is the degree of trial function in the displacement formulation(herein, p is 1 since linear elements are used in this report).

In **feap**, $\hat{\sigma}_{ij}$ can be obtained by using the macro command

stre ,node

and *eproj* can be obtained by using the macro command

erro.

As the **feap** did not have the capability to output $(verror)_i$, it was slightly modified to send the values of $\xi_i = (verror)_i/e_m$ to the standard output file.

3.3 Evaluation of local parameters

The local parameters h ,s , and α need to be defined at every point within the solution domain. Therefore, the nodal projection of each h_i has to be calculated based upon the updated edge length h_i' which is described in the previous section and evaluated with a point M whose coordinates are x_M ,y_M . The nodal projection of h_i is calculated by using a node averaging method. A node spacing h at a point in the element can be evaluated by interpolating the nodal values of h_i using the shape function N_i . Thus for the natural coordinates ξ,η associated with a point, h can be calculated as

$$h = \sum_i N_i h_i \tag{3.6}$$

To evaluate the stretching parameter s at the same point, the origin of the co-ordinate system is first moved to the point. Next, the x-y co-ordinate system is transformed to the polar co-ordinate one:

$$x = x_M + r\ cos(\theta), y = y_M + r\ sin(\theta) \tag{3.7}$$

Then the stretching parameter is evaluated as the maximum of s:

$$s = \frac{\partial h}{\partial r} \tag{3.8}$$

and the vector $\bar{\alpha}$ is determined as the direction which gives the maximum values for s . That is to say, the angle θ of the vector α which satisfies

$$\frac{\partial s}{\partial \theta} = 0. \tag{3.9}$$

On the other hand, $\partial h/\partial r$ and $\partial h/\partial \theta$ are evaluated from $\partial h/\partial x$,$\partial h/\partial y$ as follows.

$$\begin{bmatrix} \frac{\partial h}{\partial r} \\ \frac{\partial h}{\partial \theta} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial r} & \frac{\partial y}{\partial r} \\ \frac{\partial x}{\partial \theta} & \frac{\partial y}{\partial \theta} \end{bmatrix} \begin{bmatrix} \frac{\partial h}{\partial x} \\ \frac{\partial h}{\partial y} \end{bmatrix} \tag{3.10}$$

By substituting eq.(3.7) into eq.(3.10),

$$s = \frac{\partial h}{\partial r} = cos(\theta)\frac{\partial h}{\partial x} + sin(\theta)\frac{\partial h}{\partial y} \tag{3.11}$$

is obtained and by substituting eq.(3.11) into eq.(3.9)

$$-sin(\theta)\frac{\partial h}{\partial x} + cos(\theta)\frac{\partial h}{\partial y} = 0 \tag{3.12}$$

or

$$tan(\theta) = \frac{\frac{\partial h}{\partial y}}{\frac{\partial h}{\partial x}}$$

is obtained. Therefore, vector $\vec{\alpha}$ can be expressed as

$$\vec{\alpha} = \left(\frac{\partial h}{\partial x}, \frac{\partial h}{\partial y} \right) \tag{3.13}$$

Furthermore, by substituting eq.(3.12) into eq.(3.11), we will get

$$|s| = \left(\left(\frac{\partial h}{\partial x} \right)^2 + \left(\frac{\partial h}{\partial y} \right)^2 \right)^{\frac{1}{2}}. \tag{3.14}$$

On the other hand $\partial h/\partial x$ and $\partial h/\partial y$ can be calculated as

$$\begin{bmatrix} \frac{\partial h}{\partial x} \\ \frac{\partial h}{\partial y} \end{bmatrix} = J^{-1} \begin{bmatrix} \frac{\partial h}{\partial \xi} \\ \frac{\partial h}{\partial \eta} \end{bmatrix} = \begin{bmatrix} x_{,\xi} & y_{,\xi} \\ x_{,\eta} & y_{,\eta} \end{bmatrix}^{-1} \begin{bmatrix} h_{,\xi} \\ h_{,\eta} \end{bmatrix} \tag{3.15}$$

where,

$$x_{,\xi} = \sum_i \frac{\partial N_i}{\partial \xi} x_i \quad y_{,\xi} = \sum_i \frac{\partial N_i}{\partial \xi} y_i$$

$$x_{,\eta} = \sum_i \frac{\partial N_i}{\partial \eta} x_i \quad y_{,\eta} = \sum_i \frac{\partial N_i}{\partial \eta} y_i$$

$$h_{,\xi} = \sum_i \frac{\partial N_i}{\partial \xi} h_i \quad h_{,\eta} = \sum_i \frac{\partial N_i}{\partial \eta} h_i$$

Thus, s and α can be calculated by substituting eq.(3.15) into eq.(3.13) and eq.(3.14) respectively.

3.4 Examples

3.4.1 Short cantilever beam

We define the relative error to use as the indicator to measure the error for the entire mesh as follows.

$$Relative\ error = \sqrt{eerror/eproj} \qquad (3.16)$$
$$eerror = \sum_i (verror)_i$$

For an optimal mesh, it is known that the convergence rate of the relative error is given by

$$O(NDF)^{-p/2} \qquad (3.17)$$

where NDF is the number of degrees of freedom and p is the polynominal order of the shape functions.

Figure 17 describes the problem and shows the relative error dependence on NDF obtained by Zienkiewicz et. al.[9] for a short cantilever beam. Figure (a) is an analysis condition used and Fig.(b) shows the results. The results obtained from an adaptive mesh refinement by the proposed quadrilateral mesh generator are denoted by the symbol × in Fig.(b). It is clear that the convergence rate is asymptotic to eq.(3.17) and that it gives slightly better results than the unstructured triangular mesh generator of Zienkiewicz et. al.

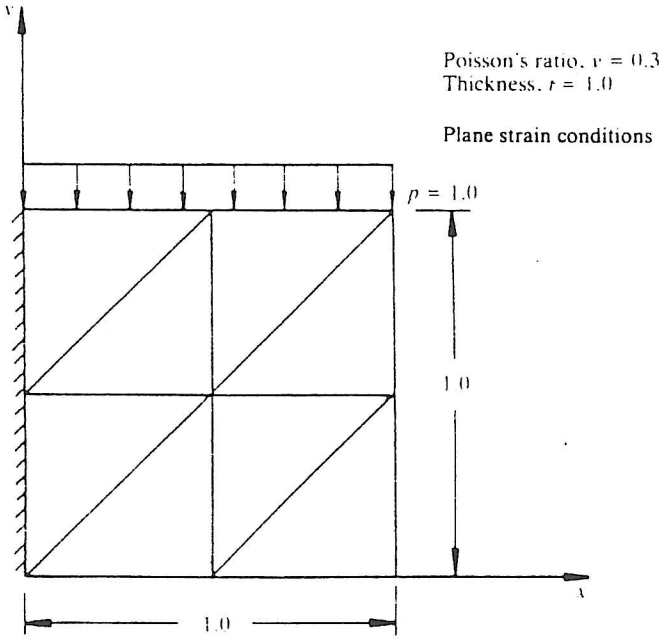
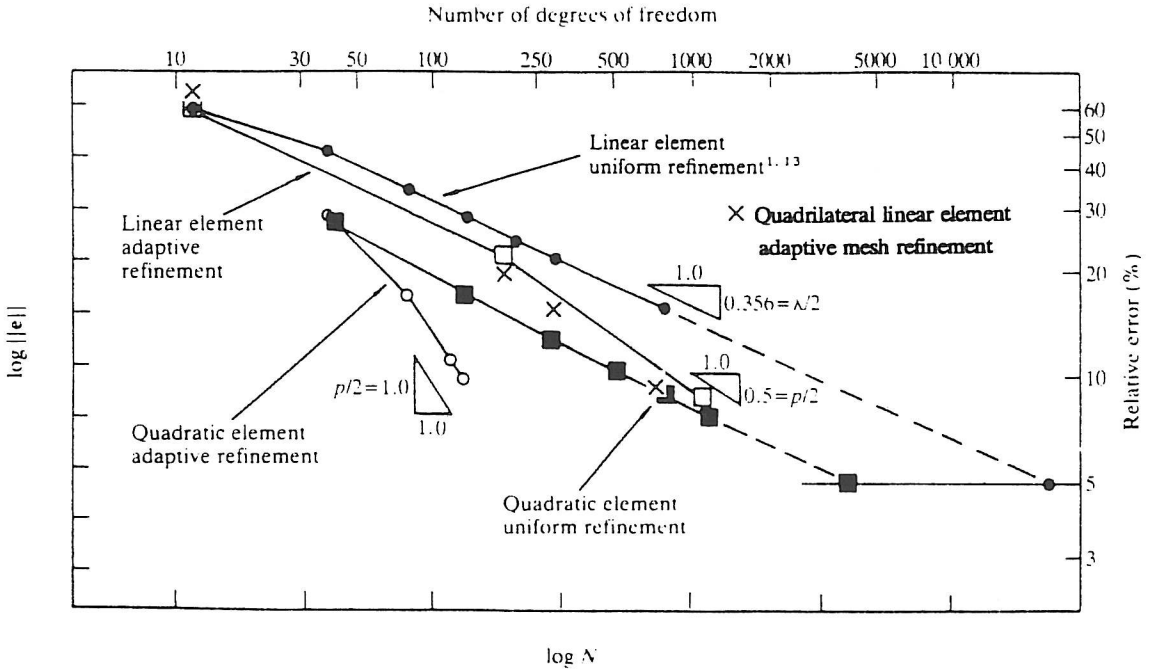


Fig. 14.4 Short cantilever beam

(a)



λ/2=0.356, theoretical rate of convergence for uniform refinement

p/2, maximum rate of convergence

Fig. 14.5 Experimental rates of convergence for short cantilever beam.

(b)

Fig.17 An application to short cantilever beam

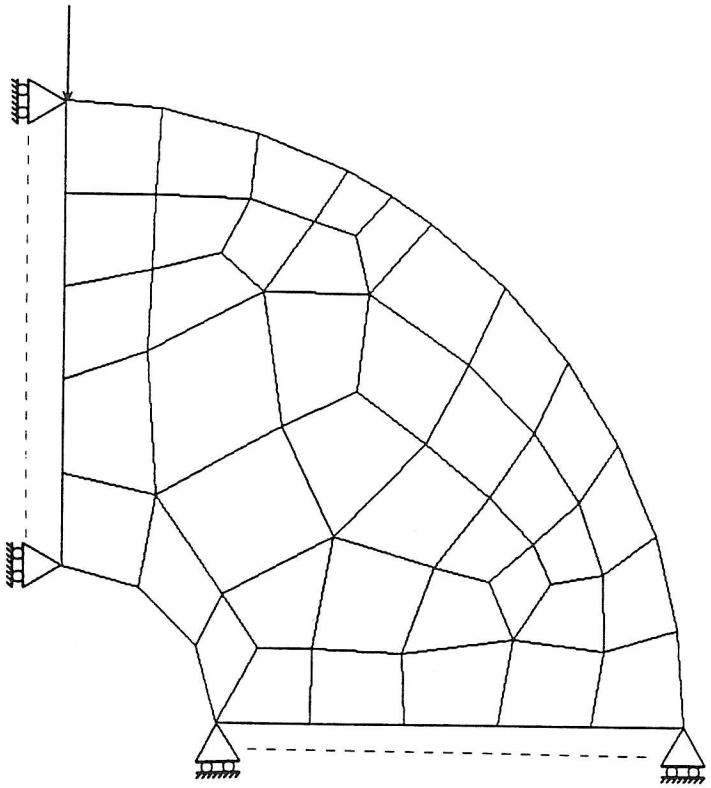
3.4.2 Stress concentration problem

The proposed algorithm is applied to an adaptive mesh refinement solution of a circular cylinder under diametrical loading. The results are shown in Fig.18. The refinement cycle starts with the initial mesh shown in Fig.(a). Then the 2nd mesh is computed according to the algorithm described above and the results are shown in Fig.(b). Finally, the 3rd mesh is constructed and it gives the results shown in Fig.(c). It is confirmed that an all-quadrilateral mesh is achieved at all steps. The enlargement of the region enclosed by the bold line in Fig.(c) is shown in Fig.(d).

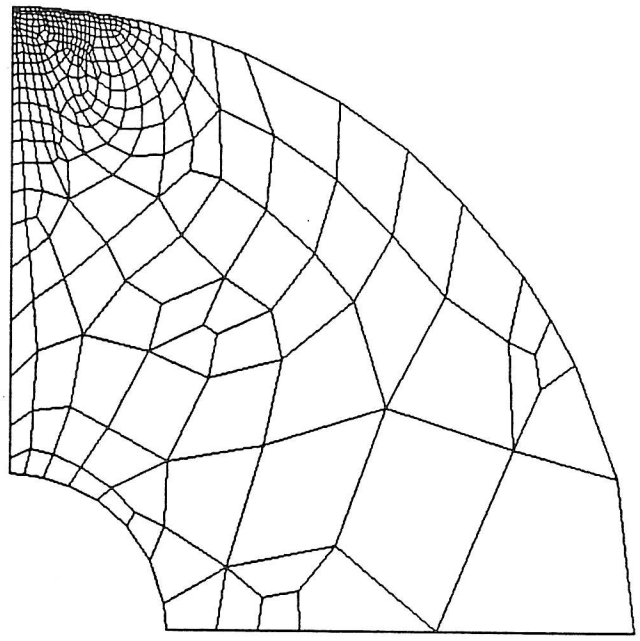
Figure 19 shows the results for a quadrant under diametrical loading. Starting with the initial mesh shown in Fig.(a), it produces the 2nd mesh shown in Fig.(b). Finally the 3rd mesh is given as shown in Fig.(c). The enlargement of the region enclosed by the bold line in Fig.(c) is shown in Fig.(d) and the further enlargement of the region enclosed by the bold line in Fig.(d) is shown in Fig.(e). It is confirmed that an all-quadrilateral mesh can be achieved for all steps. Therefore, it seems that the present method is applicable even in the regions around singular points.

Figure 20(a)(b) show the results from FE analysis corresponding to Fig.19(a)(c) respectively. These results plot contour lines of maximum shear difference which correspond to the isochromatic lines in photoelasticity. Figure (c) shows the enlargement of Fig.(b) which corresponds to Fig.19(d). Fig.(d) shows the results from a photoelastic experiment. It is apparent that the contour lines match the isochromatic lines well even at the region around the singular point (see Fig.(c)).

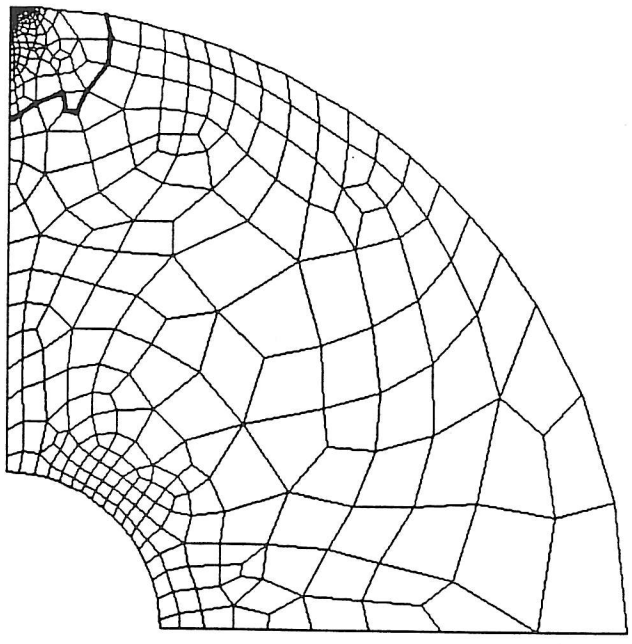
Figure 21 shows the result when the present method is applied to adaptive mesh refinement of a solution domain with a complicated multi connected boundary geometry. The initial mesh was already shown in Fig.11(b) and Fig.(a) shows the 2nd mesh. The enlargement of the region enclosed by the bold line in Fig.(a) is shown in Fig.(b). This confirms that an all-quadrilateral mesh is achieved. Thus, the present method is even applicable to mesh refinement of domains with complicated boundary geometry.



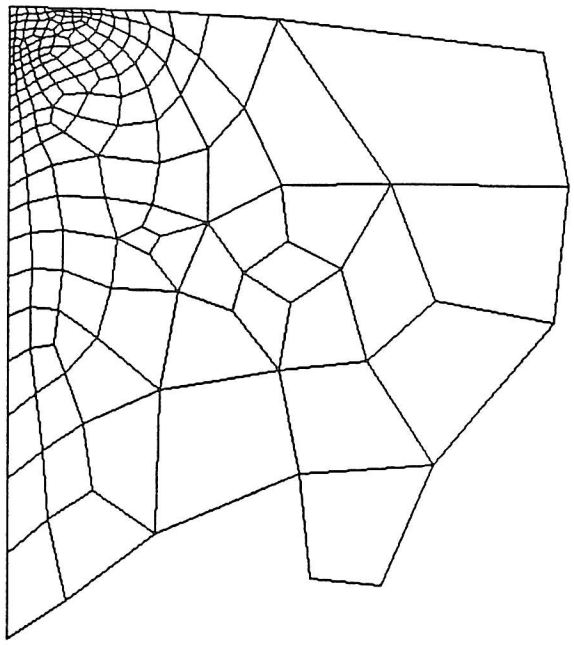
(a) Initial mesh



(b) 2nd mesh

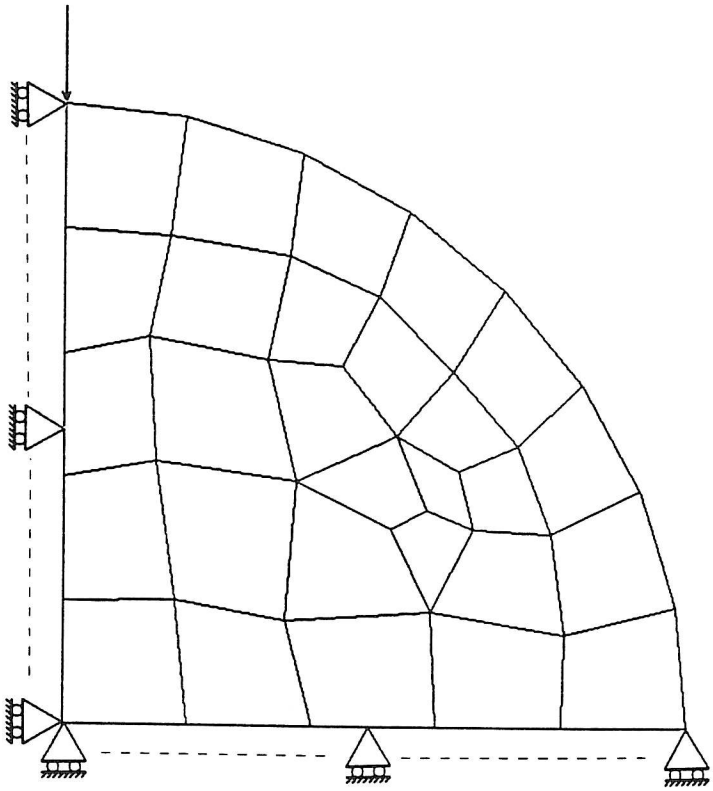


(c) 3rd mesh

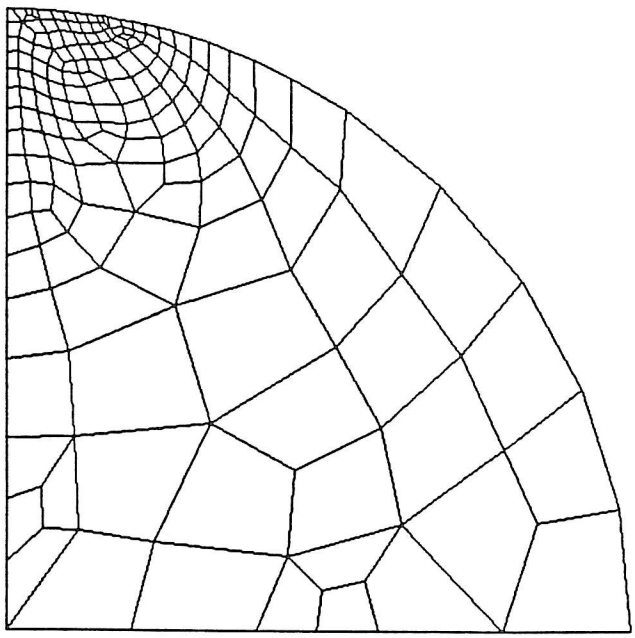


(d)Enlargement of the region enclosed by the bold line in fig.(c)

Fig.18 Cylinder under diametrical loading

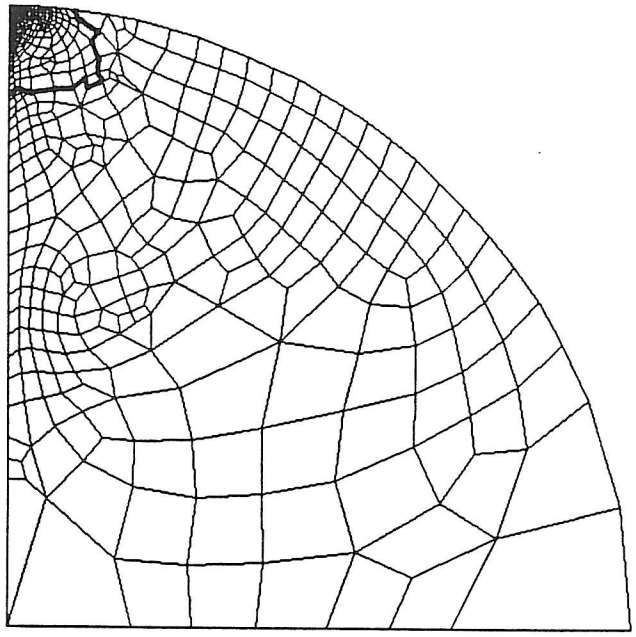


(a) Initial mesh

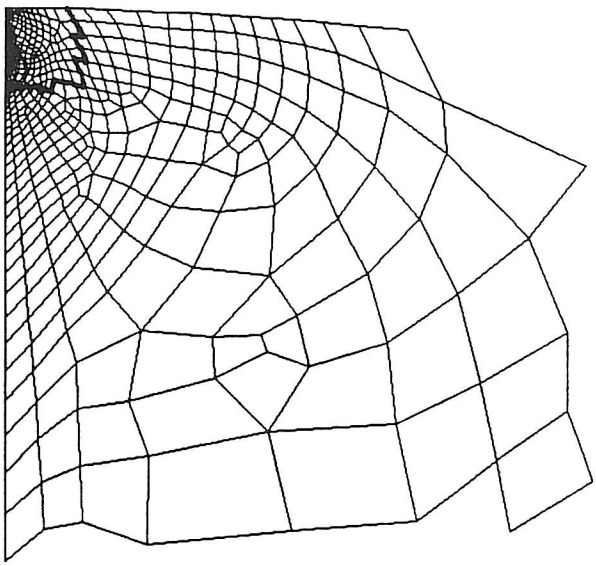


(b) 2nd mesh

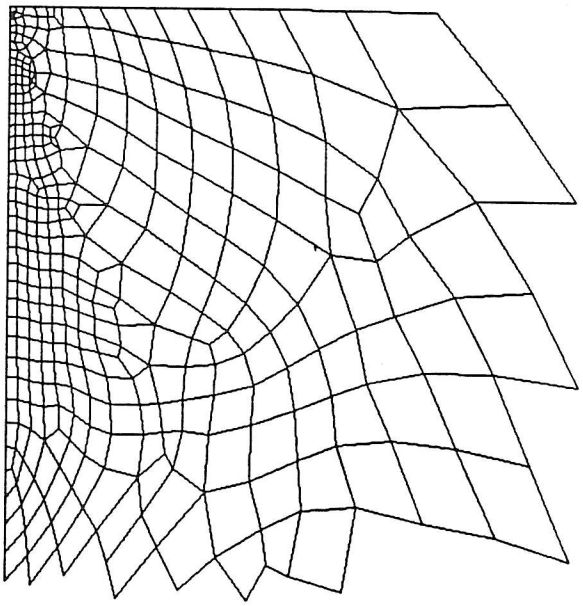
Fig.19 Quadrant disk under diametrical loading



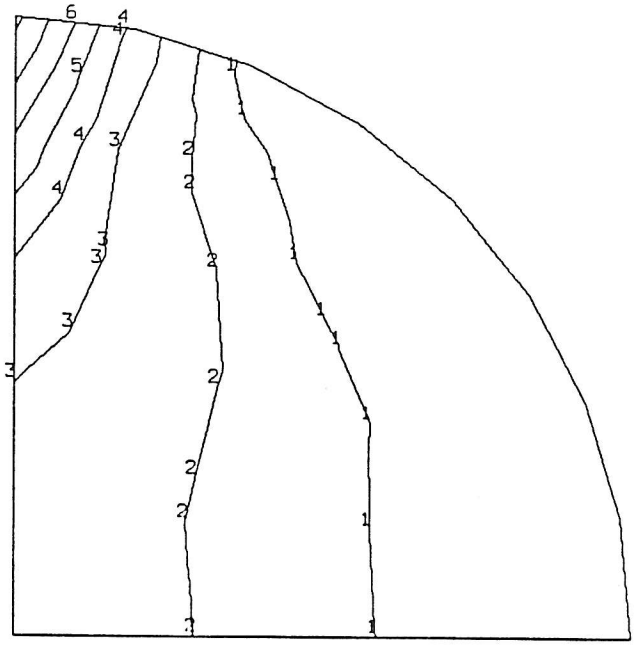
(c) 3rd mesh



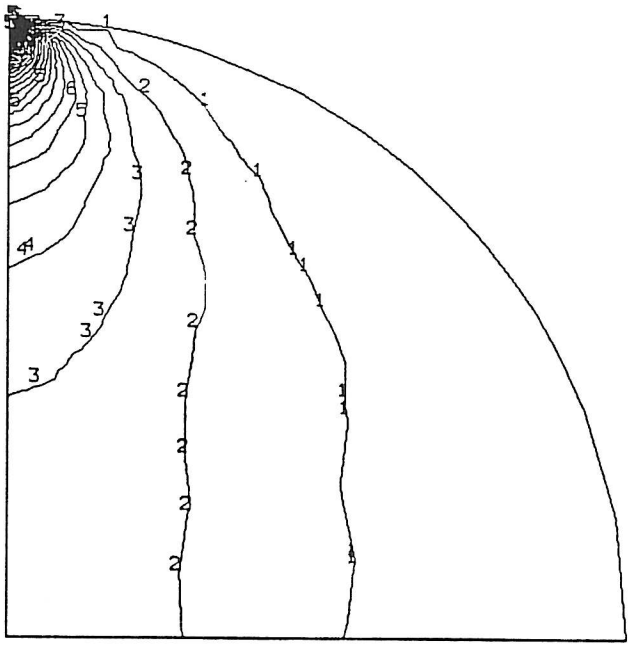
(d) Enlargement of the region enclosed by the bold line in fig.(c)



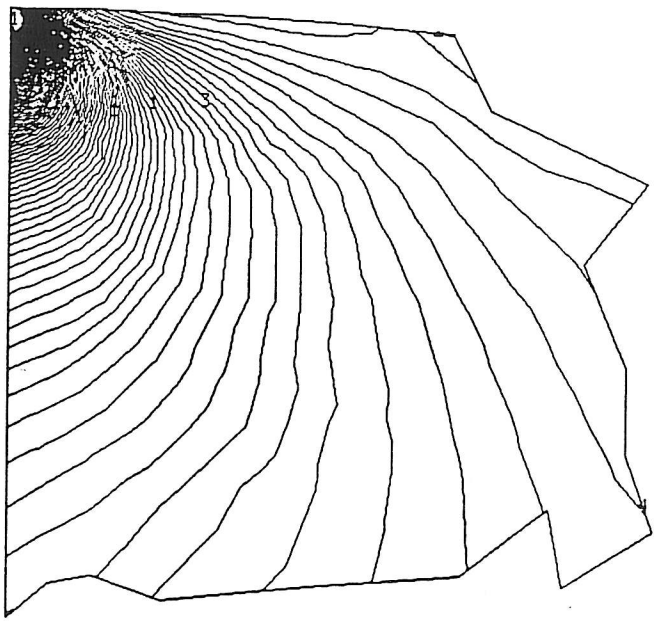
(e) Enlargement of the region enclosed by the bold line in fig.(d)



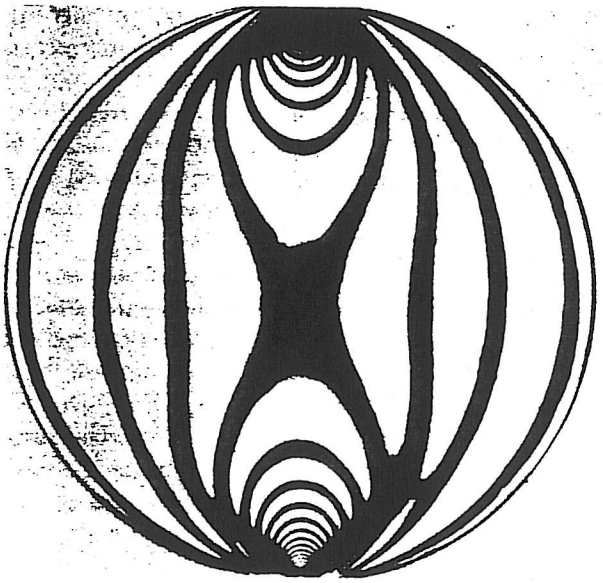
(a) Coarse mesh



(b) Refined mesh

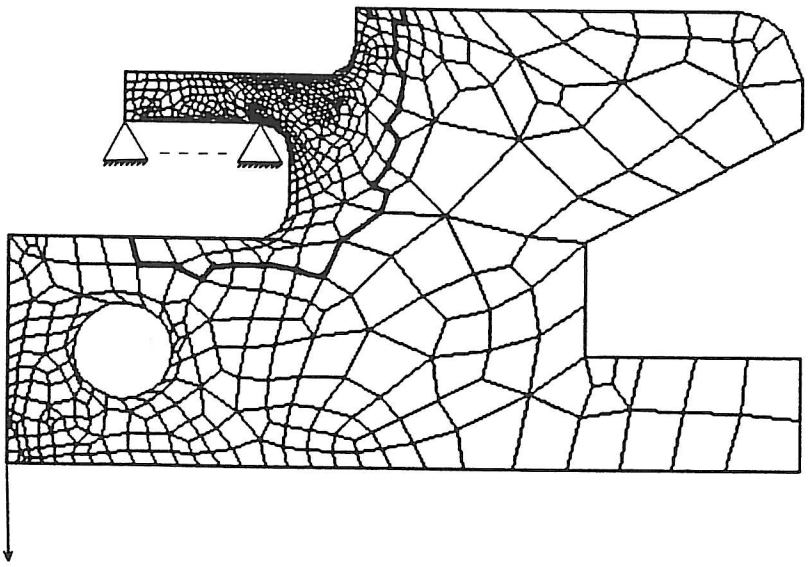


(c) Enlargement around the singular point

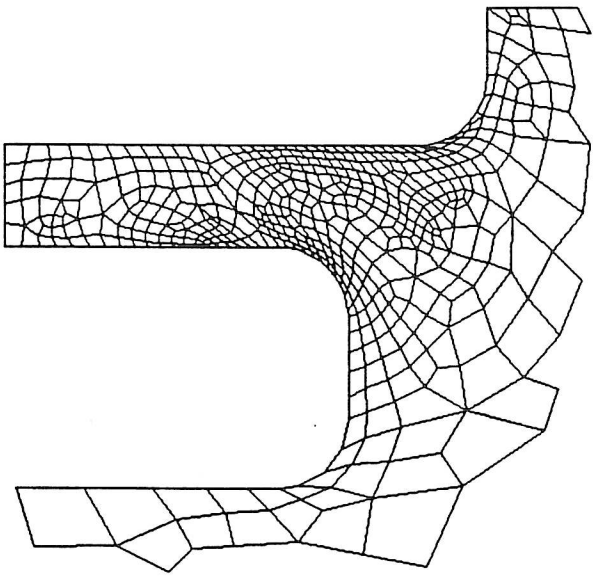


(d) Isochromatic lines in photoelasticity

Fig.20 Comparison of the contour calculated from the refined mesh with the isochromatic lines in photoelasticity



(a) 2nd mesh after refinement

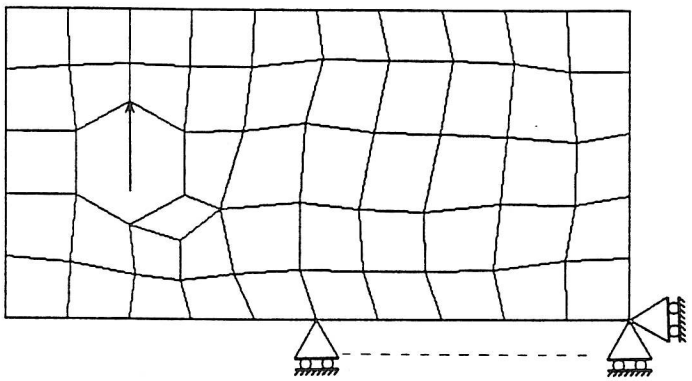


(b) Enlargement of the region
enclosed by bold line in fig.(a)

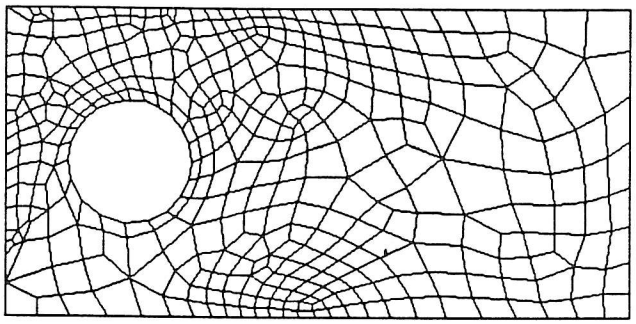
Fig.21 Adaptive mesh refinement when applied to complicated boundary domain

3.4.3 The crack problem

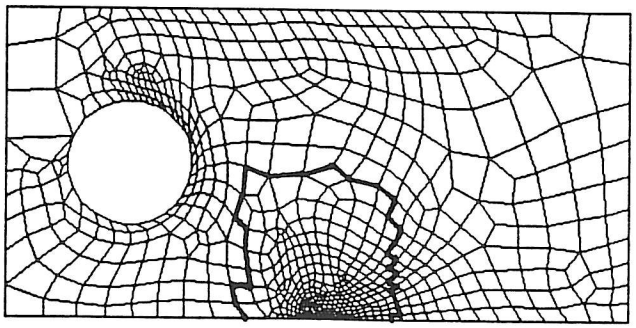
Figure 22 presents the results of an adaptive mesh refinement for a crack problem. The specimen type considered is CT(Compact Tension). Figure (a) shows the initial mesh and the boundary conditions used. The 2nd and the 3rd mesh are shown in Fig.(b) and Fig.(c), respectively. The enlargement of the region enclosed by the bold line in Fig.(c) is shown in Fig.(d). Thus, an all-quadrilateral mesh is achieved even at the region around the singular point at the crack tip.



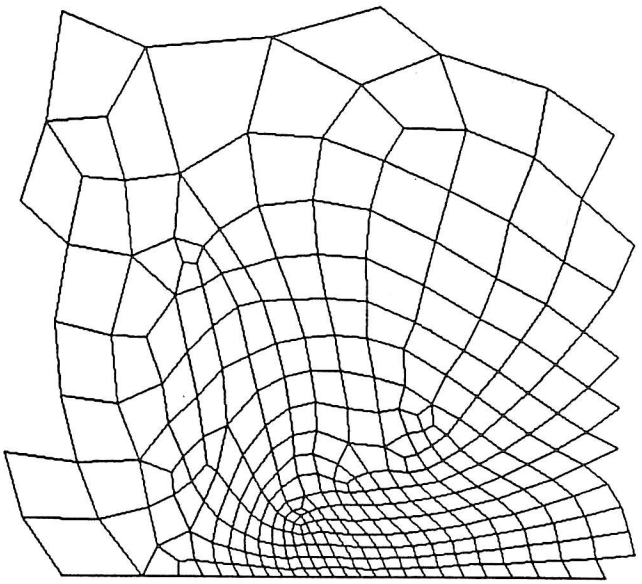
(a) Initial mesh



(b) 2nd mesh



(c) 3rd mesh



(d) Enlargement of the region enclosed by the bold line in fig.(c)

Fig.22 The crack problem(CT specimen)

4. Conclusions

A new algorithm for an unstructured quadrilateral mesh generator is presented. The algorithm achieves all-quadrilateral meshes by extending the concepts used by Zienkiewicz et. al.[1] This method has greater generality than other methods because there is no restriction on the boundary geometry and the processing is automatic even for complicated boundary geometries. Applying the present method to the case where edge lengths are exponentially distributed confirms that it is applicable even to high gradient regions. This method required only that the edge length be defined at every point within the region. Therefore, the generation method is easy to apply to adaptive mesh refinement and modeling of experiments. Some elastic stress problems have shown that the method gives satisfactory results for both stress concentration and crack problems.

Further work is needed to develop an algorithm which improves the shape of the generated quadrilateral mesh.

REFERENCES

1) J. Peraire, M. Vahdati, K. Morgan, and O. C. Zienkiewicz, 'Adaptive remeshing for compressible flow computations', *J. Computational Physics*, **72**, 449-466(1987)

2) P. L. Baehmann, S. L. Wittchen, M. S. Shephard, K. R. Grice, and M. A. Yerry, 'Robust, geometrically based, automatic two dimensional mesh generation', *Int. J. Numer. Methods Engng.*, **24**, 1043-1078(1987)

3) F. Cheng, J. W. Jaromczyk, J. Lin, S. Chang, and J. Lu, 'A parallel mesh generation algorithm based on the vertex label assignment scheme', *Int. J. Numer. Methods*, **28**, 1429-1488(1989)

4) S. H. Lo, 'A new mesh generation scheme for arbitrary planar domains', *Int. J. Numer. Methods Engng.*, **21**, 1403-1426(1985)

5) M. A. Yerry,'Modified-quadtrees for finite element mesh generation', Master's Thesis,*Center for Interactive Computer Graphics*, RPI, Troy, NY(1982)

6) M. Hoit and E. L. Wilson, 'An equation numbering algorithm based on a minimum front criteria', *Computer & Structures*, **16**, 225-239(1983)

7) I. Babuška and W. C. Rheinboldt, 'Adaptive approaches and reliability estimations in finite element analysis', *Computer Meth. Appl. Mech. Engng.*, **17/18**, 519-540(1979)

8) E. Hinton and J. S. Campbell, 'Local and global smoothing of discontinuous finite element functions using a least squares method', *Int. J. Numer. Methods Engng.*, **8**, 461-480(1974)

9) O. C. Zienkiewicz and R. L. Taylor, 'The Finite Element Method', 4th ed., vol.1, McGraw-Hill(1989)