

# QUADRILATERAL MESHING WITH ANISOTROPY AND DIRECTIONALITY CONTROL VIA CLOSE PACKING OF RECTANGULAR CELLS

Naveen Viswanath<sup>1</sup>, Kenji Shimada<sup>2\*</sup>, Takayuki Itoh<sup>3</sup>

<sup>1</sup> Carnegie Mellon University, Pittsburgh, PA, U.S.A. <naveen@biosys.net>

<sup>2</sup> Carnegie Mellon University, Pittsburgh, PA, U.S.A. <shimada@cmu.edu>

<sup>3</sup> IBM Research, Tokyo Research Laboratory, Yamato, Kanagawa, Japan. <itot@computer.org>

## ABSTRACT

A new technique for automatically generating anisotropic quadrilateral meshes is presented in this paper. The inputs are (1) a 2-D geometric domain and (2) a desired anisotropy – defined as a metric tensor over the domain – specifying mesh sizing in two independent directions. Node locations are obtained by closely packing rectangles in accordance with the inputs. The centers of the rectangles, or node, are then connected using anisotropic Delaunay triangulation that takes into account the desired anisotropy. The obtained triangular mesh is converted into a quadrilateral mesh using mesh conversion templates. The novelty of the method is that closely packed rectangles resemble a pattern of Voronoi polygons corresponding to a well-shaped quadrilateral mesh. The result is a high quality mesh that conforms well to the input. As a sample application, this method was used to generate a mesh to solve a steady state heat transfer problem.

**Keywords:** mesh generation, anisotropy, computational geometry, quadrilateral, circumcircle test.

## 1. INTRODUCTION

Mesh generation is used in a variety of areas such as finite element method and computer graphics. Today, such computational techniques form an integral part of design and analysis. Mesh generation using quadrilateral elements is computationally expensive due to constraints on element size and shape, mesh directionality control and adaptive remeshing capabilities. The problem is further restricted by specifications on anisotropy. Therefore a high quality, automatic mesh generation algorithm can increase productivity by reducing the time spent on generating the mesh.

This paper describes a new computational automated method by which a two-dimensional domain can be meshed into anisotropic quadrilateral elements. It has been shown that quad elements perform better in the FEM analysis of plane stress [1]. They are also preferred in computational fluid dynamics, sheet metal bending, and automobile crash simulation. In addition, an anisotropic mesh aligned to specific directions is better in terms of computational cost and solution accuracy when the physical phenomenon being analyzed has a strong directionality or when the material properties are anisotropic. For example, analyses involving shock propagation or analysis of fiber-reinforced glass or plastics benefit from an anisotropic mesh.

Our method to generate an anisotropic quad mesh is an extension of the bubble mesh method [2, 3] in which bubbles or spheres are packed to obtain node locations suitable for an isotropic triangular mesh. Later ellipsoids were packed to

generate anisotropic triangular meshes[4]. Square packing was done to obtain isotropic quadrilateral meshes[5]. In the current method, rectangular cells are packed to obtain the desired anisotropy. This method is readily extensible to three dimensions by packing parallelepipeds instead of rectangles.

The advantage of this approach is that node spacing and mesh directionality can be precisely controlled, independent of the boundary edges. This can be important to capture physical phenomena such as high stress gradients in stress analysis or shocks in fluid flow. Locally, the size and orientation of the rectangles are adjusted based on the input sizing and direction information. This results in a quadrilateral mesh that is anisotropic, well-shaped and well-aligned.

Another advantage is that adaptive remeshing is not computationally expensive because dynamic simulation can be continued from an existing mesh, instead of starting from scratch.

As a demonstration of the capabilities of this technique, a steady state heat transfer problem was solved using a mesh generated by our method. Since the central theme of this paper is anisotropic quadrilateral mesh generation, this is described separately in section 6.

## 2. PROBLEM STATEMENT

The problem addressed in this paper can be stated as follows

Given:

1. a two dimensional geometric domain

---

\* Correspondence to: Kenji Shimada, Mechanical Engineering, Carnegie Mellon University  
5000 Forbes Avenue, Pittsburgh, PA 15213-3890, Phone: (412) 268-3614, Fax: (412) 268-3348

2. a desired mesh anisotropy - element sizing and mesh directionality given as a 2x2 tensor field  $\mathbf{M}$

Obtain:

a well-shaped, graded, anisotropic quadrilateral mesh conforming to the input anisotropy – specified by node spacing and mesh directionality.

### 3. PREVIOUS WORK

There have been several reviews of mesh generation algorithms [6-9]. Recently surveys of mesh generation algorithms and related software have been published on the world wide web [10-12].

Quadrilateral meshing has been implemented using a variety of techniques. One common method is node placement followed by connection. This is popular due to the existence of a robust scheme for connection called Delaunay Triangulation. The obtained triangular mesh is then converted into a quadrilateral mesh, and many such conversion methods have been proposed [13-16]. Other approaches have also been published. A CSG-based approach for node placement was proposed by Lee [17, 18]. In paving [19] proposed by Blacker and Stephenson, quadrilateral elements are created one by one, starting from the boundary. The Q-Morph algorithm proposed by Owen [20] works similarly, converting a triangular mesh to a quadrilateral mesh starting from the boundary. In these advancing front methods, mesh directionality cannot be controlled independent of the boundary, which may be needed to reflect load conditions or material properties.

Castro-Diaz et al. showed the use of a metric tensor to improve the quality of adapted meshes in flow computations [21]. Borouchaki et al. demonstrated the use of a metric tensor to generate an anisotropic triangular mesh [22]. Bossen and Heckbert used a 2x2 metric tensor to generate an anisotropic triangular mesh using a system of interacting particles [23]. Shimada et al. used ellipsoid packing to obtain anisotropic triangular meshes [4]. The biting ellipses scheme [24] combines paving and packing to generate an anisotropic triangular mesh. Though this method has a theoretical time bound for convergence, meshing even simple geometries is lengthy.

Our work draws on both isotropic quadrilateral meshing techniques and anisotropic triangular meshing methods and incorporates new techniques to realize efficient, high-quality anisotropic quadrilateral meshing.

### 4. ANISOTROPIC QUADRILATERAL MESH GENERATION

#### 4.1 Outline of Technical Approach

The anisotropic quadrilateral meshing problem is solved in the following way.

- Step 1 : Place rectangular cells on all vertices

- Step 2 : Pack rectangular cells on all edges

- Step 3 : Pack rectangular cells on the faces

- Step 4 : Place nodes at the centers of the rectangles

- Step 5 : Generate triangular mesh topology using anisotropic Delaunay Triangulation

- Step 6 : Selectively combine pairs of triangles to obtain a quad-dominant mesh

- Step 7 : Use mesh conversion templates to obtain an all-quad mesh

Figure 1 illustrates this process. The first three steps generate suitable node locations by closely packing rectangles in accordance with the given input. The sizes and directions of the rectangles are adjusted based on the given mesh sizing and mesh directionality information.

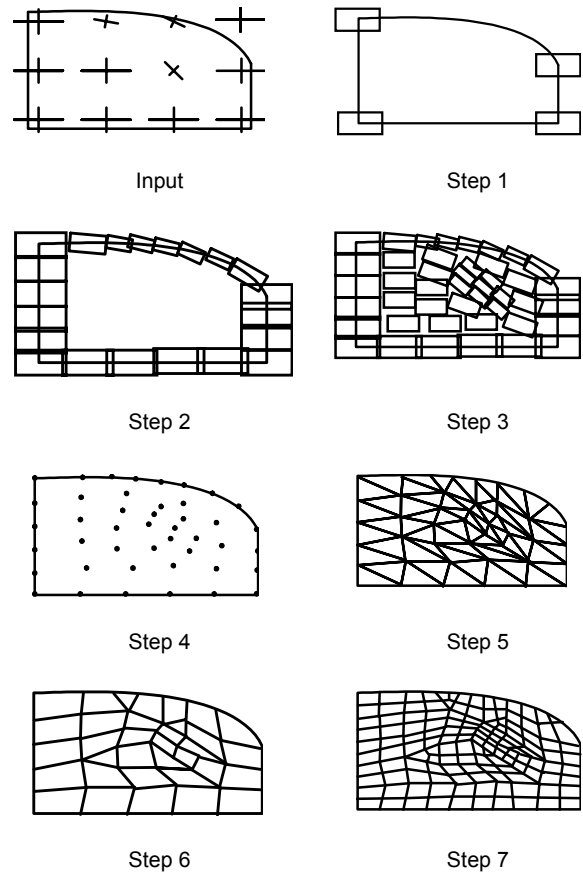
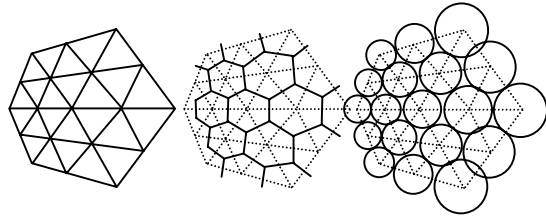


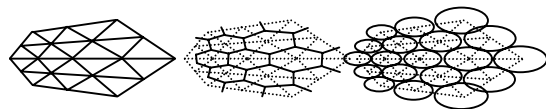
Figure 1 The mesh generation process

Figure 2 shows various meshes and their Voronoi polygons. Figure 2(a) shows a set of packed circles that mimics Voronoi polygons of a well-shaped triangular mesh [2,3]. By packing a set of ellipses instead of circles, as shown in Figure 2(b), Voronoi polygons of a well-shaped anisotropic

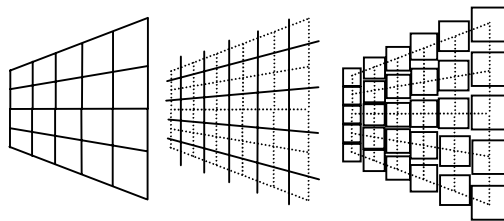
triangular mesh can be mimicked [4]. Figure 2(c) shows a set of packed squares that mimics Voronoi polygons of a well-shaped quadrilateral mesh [5]. This paper demonstrates a technique for mimicking Voronoi polygons of a well-shaped anisotropic quadrilateral mesh by packing a set of rectangles instead of squares, as shown in Figure 2(d).



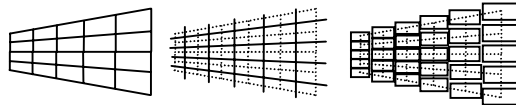
(a) Triangular mesh, Voronoi polygons, and packed circles.



(b) Anisotropic triangular mesh, Voronoi polygons, and packed ellipses.



(c) Quadrilateral mesh, Voronoi polygons, and packed squares.



(d) Anisotropic quadrilateral mesh, Voronoi polygons, and packed rectangles.

**Figure 2 Voronoi Polygons and Meshes**

Some of the issues that arise are as follows:

1. What are the optimal locations of the rectangles?
2. How many rectangles are needed to fill the domain?
3. How to modify Delaunay triangulation to produce anisotropic triangles?
4. How to convert the obtained triangles to quads?

For the first issue we use a physically-based model, similar to a particle system in computer graphics. A proximity-based force field is defined between two rectangles so that either an attracting force or a repelling force is applied based on the distance between them. The equations of motion are solved numerically assuming a point mass at the center of the rectangles and viscous damping.

The second issue is resolved by checking the local population density and adaptively adding or removing rectangles during dynamic simulation of the motion of the rectangles. After an iteration, if a region has an excessive number of rectangles, then rectangles are removed, and if there are significant holes, rectangles are added.

Delaunay triangulation tries to locally maximize the minimum angle in pairs of triangles, which ideally produces equilateral triangles. In anisotropic meshing, however, long, slender triangles may be preferred. So, the Delaunay circumcircle test is suitably modified using the input metric tensor to evaluate which of the two possible triangulations of four nodes is desirable.

Once the triangular mesh is obtained, it is first converted to a quad-dominant mesh using the input directionality data, and then to an all-quad mesh using mesh conversion templates. Each of these topics is addressed in the following subsections.

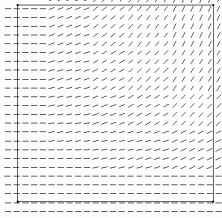
#### 4.2 Mesh Anisotropy – Specification and Interpretation

An input 2x2 metric tensor field is used to specify the desired anisotropy. As in previous work [4, 21-23] this tensor is symmetric and positive-definite. The following topics are explained:

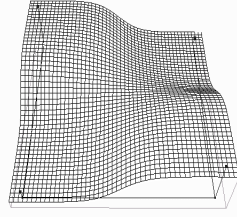
1. How to calculate values of element sizes and directions, given a metric tensor
2. How to specify the metric tensor for a required anisotropy

Mesh directionality, mesh sizing and therefore, anisotropy can be specified in a single compact form using tensor notation. Anisotropy implies that one edge of an element has a significantly different length than the other(s) – resulting in elements that are stretched. Though these three concepts are inter-related, it is important to understand that anisotropy cannot be obtained using mesh directionality control and mesh sizing control together. We can obtain anisotropy only if the sizing can be controlled independently in different directions. This is different from simply using isotropic quadrilaterals and orienting them along a given direction.

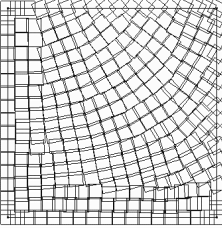
The meshes in Figure 3 all have the same input geometry and mesh directionality. Figure 3(a) and Figure 3(b) show the input geometry, mesh directionality, and node spacing function. In Figure 3(c), uniform squares are packed in the domain according to specified mesh directions, and an isotropic quadrilateral mesh is then generated, as shown in Figure 3(d). Even if the sizes of the squares are varied as shown in Figure 3(e), node locations suitable for a graded quadrilateral mesh are obtained, but the mesh shown in Figure 3(f) is still isotropic. By virtue of packing rectangles with specified aspect ratios as shown in Figure 3(g), an anisotropic quadrilateral mesh is obtained by controlling the node spacing in two different directions, as shown in Figure 3(h). If the sizes of rectangles are varied as shown in Figure 3(i), a graded anisotropic mesh is generated as shown in Figure 3(j).



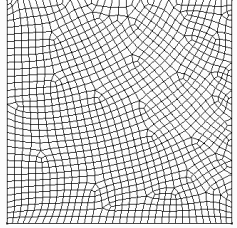
(a) Input geometry and mesh directionality



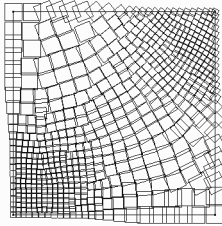
(b) Input geometry and node spacing function



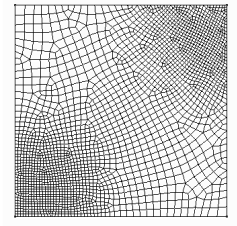
(c) Square Packing



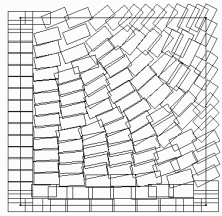
(d) Isotropic quadrilateral mesh



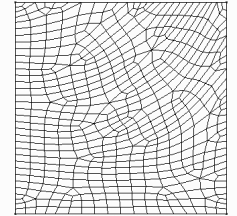
(e) Graded square packing



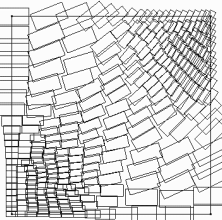
(f) Graded isotropic quadrilateral mesh



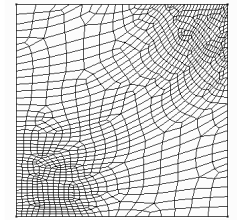
(g) Rectangle packing



(h) Anisotropic quadrilateral mesh



(i) Graded rectangle packing



(j) Graded anisotropic quadrilateral mesh

Figure 3 Square Packing vs. Rectangle Packing

The procedure to calculate the required anisotropy from a given metric tensor is described first. Following that, the method used to specify the metric tensor, given an anisotropy is explained.

Given a 2x2 tensor  $\mathbf{M}$ , we obtain the two eigen values  $\lambda_i$  by solving [25]

$$|\mathbf{M} - \lambda \mathbf{I}| = 0 \quad (1)$$

where  $\mathbf{I}$  is the 2x2 identity matrix. Once the eigen values are found, the eigen vectors  $\mathbf{x}_i$  are found using

$$\mathbf{M}\mathbf{x}_i = \lambda_i \mathbf{x}_i, \quad i = 1, 2 \quad (2)$$

The eigen vectors are the directions of the major and minor axes specifying the mesh directionality, and the eigen values are the inverse of the squares of the major and minor radii.

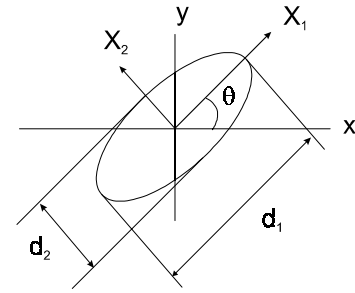


Figure 4 Specification, interpretation of a tensor as an ellipse

Analogously, given the directions and magnitudes of the axes (the desired mesh sizing and directionality as described above), the tensor can be calculated using

$$\mathbf{M} = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^T \quad (3)$$

$$\mathbf{R} = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \quad (4)$$

$$\mathbf{\Lambda} = \begin{pmatrix} \left(\frac{2}{d_1}\right)^2 & 0 \\ 0 & \left(\frac{2}{d_2}\right)^2 \end{pmatrix} \quad (5)$$

As shown in Figure 4,  $\theta$  is the angle between the major axis of the ellipse and the positive x-axis, and  $d_1$ ,  $d_2$  are the major and minor axis diameters.

In our implementation, we specify this by

1. a direction  $\mathbf{X}_2$  specifying the minor axis
2. a base size along  $\mathbf{X}_2$
3. an aspect ratio

The single direction specifies the orientation of the minor axis. Since the principal axes are perpendicular to each other, the direction of the major axis can be easily calculated. Mesh sizing information is given as a base size along the minor axis and an aspect ratio – the ratio of the diameter of the major axis to that of the minor axis. The specified “stretching” can easily be visualized using this aspect ratio.

### 4.3 Close Packing of Oriented Rectangular Cells

To get appropriate node locations, rectangles are packed closely according to a given input. A physically-based particle simulation approach is used to solve this rectangular packing problem efficiently.

An input metric tensor field is used to specify the length, breadth and orientation of quadrilateral elements in the domain. This data is stored in a background grid at discrete locations. At intermediate points, the data at the four grid nodes enclosing the point are linearly interpolated. Alternatively, if input is given only at a few points, then values at all nodes are calculated using a Laplacian smoothing [26] type approach to obtain a smoothly varying field.

In the triangular bubble mesh technique [2, 3] a force similar to van der Waals force produces hexagonal packing of bubbles. This field produces an attractive force if two particles are farther apart than a stable distance, and a repulsive force if they are located closer together than a stable distance. Node locations obtained this way create a well-shaped isotropic triangular mesh. This field was modified in square packing to obtain node locations suitable for isotropic quadrilateral mesh generation [5] by adding four sub-fields to the main force field. In our proposed method, the field used for square packing has been “stretched” using the input aspect ratio to obtain rectangular packing.

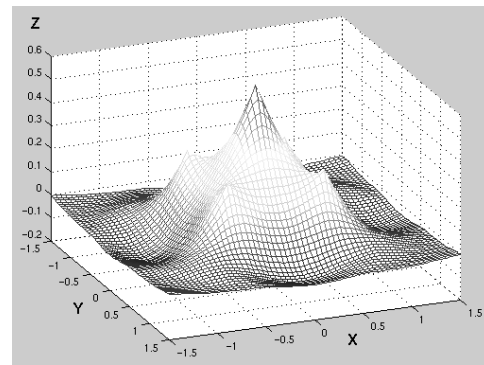
Figure 5 (a) shows the force field used in square packing. In 5 (b) the given aspect ratio is locally constant and is along the x-axis. The corresponding potential function can be obtained by integrating the force. The rectangles try to occupy minimum energy positions in this field. Packing is complete when the geometry is covered sufficiently, without any significant gaps or overlaps.

Ellipses can be used to achieve rectangle packing as shown in Figure 6. After the central rectangle, other stable positions for rectangles are the along the directions  $d_1$  and  $d_2$ . Once these are also occupied, the ones in between can be filled. For a graded anisotropic quadrilateral mesh (the base size of the rectangles and the aspect ratio may vary), the fields and the sub-fields have to be suitably adjusted to obtain packing

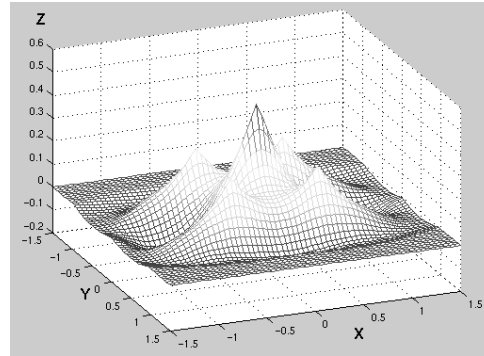
of rectangles of desired sizes which would in turn give us a mesh consistent with the input.

Using this proximity-based force, a physically-based relaxation method is used to find a closely packed configuration of rectangles. Due to the non-linear nature of the force and complicated geometric constraints, force equilibrium equations become highly non-linear and are difficult to solve using a multi-dimensional root finding technique such as the Newton-Raphson method.

The solution to this problem, instead, is to assume a point mass  $m$  at the center of each rectangular cell and a viscous damping  $c$  and solve these equations of motion (Equation 6) using a numerical integration scheme like the fourth-order Runge-Kutta method. The positions of rectangles  $\mathbf{x}_i$  are thus obtained.



(a) Force field used for square packing



(b) Force field used for rectangle packing

Figure 5 Force fields used for packing

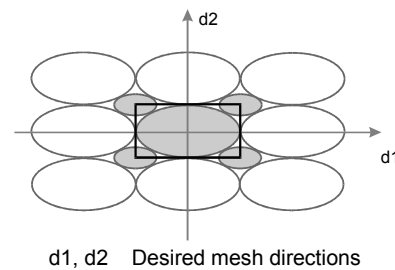


Figure 6 Rectangle packing using ellipses

$$m \ddot{\mathbf{x}}_i(t) + c \dot{\mathbf{x}}_i(t) = \mathbf{f}_i(t) \quad i = 1, 2, \dots, n \quad (6)$$

While solving this equation numerically, we adaptively adjust the number of rectangles in the domain, called “adaptive population control.” This is necessary as the number of rectangles needed for packing is unknown at the start. We generate an initial configuration using octree subdivision. During simulation, we use a procedure of adding rectangles in sparse areas and deleting rectangles in dense areas to get closely packed rectangles. This dynamic simulation and adaptive population control approach makes adaptive remeshing very efficient because we can simply continue the simulation process from previous node locations, without starting from a totally fresh configuration, when the geometry, node spacing or mesh directionality is changed.

#### 4.4. Anisotropic Delaunay Triangulation

Once a force-balancing configuration of rectangles is obtained, the centers of the rectangles must be connected to form a complete triangular mesh, which is then converted into a quad mesh. In connecting nodes, Delaunay triangulation is considered suitable for finite element analysis, as the triangulation maximizes the smallest angles of the triangles. Ideally, it creates triangles as equilateral, or isotropic, as possible for a given set of points; thus thin, or anisotropic triangles are avoided whenever possible.

One important property of Delaunay triangulation is that a circumcircle of a Delaunay triangle must not contain other nodes inside it. Many Delaunay triangulation algorithms use the circumcircle test. This test is also used in Sloan's algorithm [27] which was implemented in the original 2-D isotropic bubble mesh. The circumcircle test is performed on a pair of adjacent triangles that forms a convex quadrilateral. Given such a set of four points, the test checks if the fourth point lies inside the circumcircle of the triangle formed by the other three points. If it does, the four points are then reconnected into the other possible configuration of two triangles. This test, however, is not suitable for anisotropic meshing. Delaunay triangulation must be modified to incorporate anisotropy in the circumcircle test.

Assuming the metric tensor is locally constant, the circumcircle test is done in normalized space. The coordinates of the four nodes under consideration are transformed so that an ellipse is mapped back to a unit circle [23].

A local average tensor can be determined by first calculating the barycenter of the four nodes and then finding the metric tensor  $\mathbf{M}$  at this barycenter<sup>1</sup>.

<sup>1</sup> Slightly different anisotropic Delaunay triangulation schemes are used by other researchers [21-23]. For example, an alternative way to take an average of four metric tensors is:

$$\mathbf{M} = \frac{\sum_{i=1}^4 \mathbf{M}(x_i)}{4}$$

$$\mathbf{M} = \mathbf{M} \left( \frac{\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3 + \mathbf{x}_4}{4} \right) \quad (7)$$

where  $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4$  are position vectors of the four nodes.

This metric tensor is then used to transform the coordinates of the four nodes. A rotational and scaling transformation is used to map the ellipse corresponding to this tensor into a unit circle. The new coordinates are given by

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \mathbf{T} \begin{pmatrix} x \\ y \end{pmatrix} \quad (8)$$

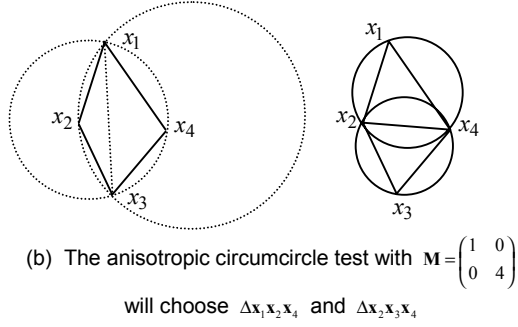
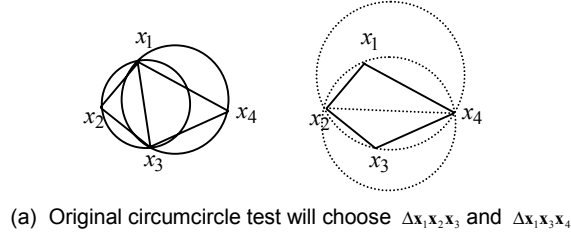
$$\mathbf{T} = \mathbf{P}\mathbf{R}^T \quad (9)$$

$(x, y)$	Coordinates of nodes
$(x', y')$	Transformed coordinates
$\mathbf{T}$	Transformation matrix
$\mathbf{R}$	$\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$
$\mathbf{P}$	$\begin{pmatrix} (2/d_1) & 0 \\ 0 & (2/d_2) \end{pmatrix}$
$d_1, d_2, \theta$	Major, Minor diameters and angle of inclination obtained from $\mathbf{M}$ , as before

The circumcircle test is then applied to the transformed normalized coordinates. This modifies the Delaunay triangulation used in the original bubble mesh program so that the input anisotropy is taken into consideration.

Figure 7 shows how a different pair of triangles is selected when the anisotropic circumcircle test is performed after transforming the positions of the four nodes. In the example shown in Figure 7, an aspect ratio of 2 is desired along the horizontal direction. The original circumcircle test 7(a) does not account for this and chooses the pair  $\Delta \mathbf{x}_1\mathbf{x}_2\mathbf{x}_3$  and  $\Delta \mathbf{x}_1\mathbf{x}_3\mathbf{x}_4$  although this does not orient the triangles according to the desired anisotropy. The modified approach first transforms the coordinates and then applies the circumcircle test. As shown in figure 7(b), this chooses the correct pair of triangles,  $\Delta \mathbf{x}_1\mathbf{x}_2\mathbf{x}_4$  and  $\Delta \mathbf{x}_2\mathbf{x}_3\mathbf{x}_4$ .

To demonstrate the effectiveness of this procedure, Figure 8 contrasts the original Delaunay triangulation and the anisotropic Delaunay triangulation. Given the same inputs, the anisotropic Delaunay triangulation creates a mesh that is stretched and “flows” along the input mesh directionality and conforms better to the desired anisotropy.



**Figure 7 Anisotropic circumcircle test**

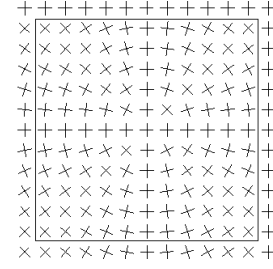
#### 4.5. Conversion to Quadrilateral Mesh

After an anisotropic triangular mesh topology is obtained, it is first converted into a quad-dominant mesh by selectively merging two triangular elements into a quadrilateral element along the given vector field. This mesh conversion algorithm consists of three stages:

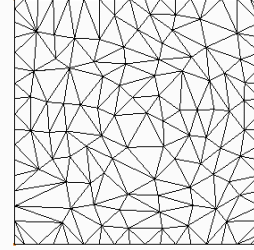
1. Calculate a score  $\Omega_i$  that measures how well the resultant quadrilateral mesh element aligns along the specified mesh directions if the  $i$ -th non-boundary edge of a triangular element is removed to form a quadrilateral.
2. Sort all the non-boundary edges using a priority queue.
3. Delete the edges successively from the top of the priority queue. The deletion of one edge creates one quadrilateral.

The score  $\Omega_i$  is calculated by comparing the directions of the four edges of the resultant quadrilateral element with the input mesh direction vectors at the centers of the edges. For side  $j$  of the quadrilateral element  $i$ , we calculate  $\omega_{ij}$  using

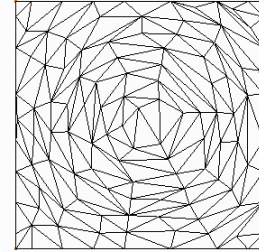
$$\omega_{ij} = \begin{cases} k_j |\mathbf{u}_{ij} \cdot \mathbf{v}_{ij}| & |\mathbf{u}_{ij} \cdot \mathbf{v}_{ij}| \geq \frac{1}{\sqrt{2}} \\ k_j \sqrt{1 - (\mathbf{u}_{ij} \cdot \mathbf{v}_{ij})^2} & |\mathbf{u}_{ij} \cdot \mathbf{v}_{ij}| < \frac{1}{\sqrt{2}} \end{cases} \quad (10)$$



(a) Input Mesh Directionality



(b) Normal Delaunay Triangulation



(c) Anisotropic Delaunay Triangulation

**Figure 8 Effect of anisotropic Delaunay triangulation**

where the subscript  $i$  is the index of a quadrilateral element and  $j = 1, 2, 3, 4$  is the index of the side edge of the element. The inner product  $\omega_{ij}$  is computed using  $\mathbf{u}_{ij}$  the unit vector of the edge  $j$  of quadrilateral  $i$ , and  $\mathbf{v}_{ij}$  the input mesh direction vector at the center of that edge.

This scoring function is similar to our previous methods [5,16], but differs because the product is weighted by  $k_j$ . For the two shorter sides of a rectangle,  $k_j$  is 1. For the longer sides  $k_j$  is the aspect ratio. The weight aligns the edges of the output quadrilateral mesh as much as possible along the primary vector.

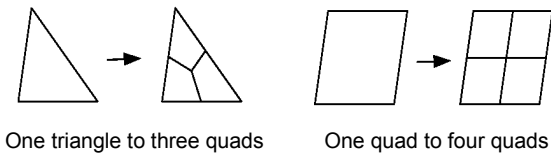
From this definition of  $\omega_{ij}$  the score  $\Omega_i$  is calculated as follows

$$\Omega_i = \frac{1}{4} \sum_{j=1}^4 \omega_{ij} \quad (11)$$

After going through the above stages, a quad-dominant mesh is obtained. This is converted into an all-quad mesh using one of two mesh conversion templates [15]

1. One triangle to three quads
2. One quad to four quads

In both these methods, a new node is inserted at the center of an element using linear interpolation of the midpoints of the edges. This node is joined to each of the applicable edges to form quadrilateral mesh elements. So, the final element size is roughly half the size of elements at the previous step.



**Figure 9 Mesh Conversion Templates**

## 5. RESULTS

This section contains the results of the automatic anisotropic quadrilateral mesh generation technique proposed in this paper. The sequence of pictures in Figures 10 and 11 is as follows. The first picture (a) is the desired mesh directionality. Picture (b) has rectangles packed in accordance with the input. The third picture is the triangular mesh obtained after anisotropic Delaunay triangulation (c). The anisotropic quadrilateral mesh is the final image (d).

The domain in Figure 10 is a circle approximated using line segments. Mesh directions shown in Figure 10(a) are those corresponding to a family of rectangular hyperbolas. The aspect ratio is varied from 2 at the left of the domain, to 4 at the right. The rectangle packing result is shown in Figure 10(b). The intermediate triangle mesh conforms to the given input, as shown in Figure 10(c). The variation in aspect ratio of quadrilateral elements is seen in Figure 10 (d).

The results shown in Figure 11 showcase the full functionality of the presented technique. Mesh directionality and sizing is independently controlled in two directions – parallel and perpendicular to the specified direction (base size of rectangle and aspect ratio). The aspect ratio is linearly varied from 2 at the right end to 3 at the left end. The base size of the rectangles is 5 at the right and increases to 10 at the left. Mesh directions are specified to coincide with the boundary. This shows how a boundary-aligned mesh may be obtained using this method, without the front collision problems typical of advancing front methods.

## 6. APPLICATION – MESH GENERATION FOR STEADY STATE HEAT TRANSFER

This section describes a procedure which can generate meshes for real world problems using the proposed method. A mesh is generated to solve a steady-state heat-transfer

problem. A preliminary analysis is used to specify the inputs required for this technique. Once the mesh is generated, it is used to solve the problem. Comparing the temperature contours of regular meshes with those of the generated mesh shows that the computational cost can be reduced without losing solution accuracy.

Figure 12(a) shows the problem to be solved. The eight points shown in the interior of the geometry are at  $300^{\circ}\text{C}$ . The four sides are at  $0^{\circ}\text{C}$ . The initial coarse mesh shown in Figure 12(b) is used for the analysis. The contour plot of the solution is shown in Figure 12(c). The directions of the temperature gradients are shown in Figure 12(d).

To specify the inputs required for the procedure, a preliminary analysis is done using a coarse mesh of 100 square elements. Even though a few important details may be missed if the mesh is too coarse with reference to the features in the problem, it is better than generating a mesh with no consideration to the boundary conditions, loads, or material properties.

Data exchange interfaces are used to exchange mesh and solution data between ANSYS and the proposed meshing scheme. The command line interface of ANSYS is used to read in scripts to import the generated mesh. The nodal solution and the list of nodes and elements in plain text format are used for export.

The flexibility of the process allows arbitrary specification of anisotropy – given as mesh sizes, aspect ratios and directions. Mesh directions are specified using temperature gradients calculated using the coarse mesh solution. In heat transfer problems, there are large temperature gradients near regions of high temperature. Thus, a small mesh size and aspect ratio is specified at regions where the high temperature is applied. This demonstrates how an expert can use this technique. A better way would be to automate this based on error bounds or other desired criteria.

The FEA code ANSYS was used to solve the problem and to make the contour plots in Figure 13. The plots all have the same scale – the same shade represents the same range of temperature in the contour plots 13(b), 13(d), and 13(f).

The mesh thus generated is shown in Figure 13(e). As a comparison, Figures 13(a) and 13(c) show a regular mesh of 900 and 1600 elements respectively. The temperature contours for the generated mesh shown in Figure 13(f) seem to be better than those for the 900 element regular mesh shown in Figure 13(b), even though it has fewer number of elements – 828. This is a qualitative comparison, assuming the 1600 element mesh has the better solution. This is a fair assumption because, in such a straightforward problem, when the number of uniform elements is increased, better solutions are obtained. For instance, the contours are smoother and the resolution of the high temperatures is better. The second contour from high temperature spots of the generated mesh 13(f) resembles those of 13(d), more than the corresponding contour of 12(b) resembles 13(d). The number of elements is a good measure of the computational cost. Since better results are obtained with fewer elements, this leads us to believe that the proposed technique results in a reduction in computational cost without loss in accuracy of analysis.



## 7. CONCLUSION AND FUTURE WORK

This paper presents a new technique for generating anisotropic quadrilateral meshes in two-dimensions, using a physically - based method for obtaining a close packing of rectangles. The novelty of this technique is that closely packed rectangles resemble a pattern of Voronoi polygons that correspond to a well-shaped, well-aligned, anisotropic quadrilateral mesh.

The centers of the rectangles give the node locations. Though a triangular mesh topology is obtained initially, it is well suited for conversion to a quad mesh because rectangles have been packed using an appropriate force-field.

A significant advantage of our technique is the ability to control the mesh sizing and direction independent of the boundary. This can be used to achieve effective adaptive remeshing because meshes can be tailored to suit specific application domains and needs – such as load conditions, or properties of materials in stress analysis. Also, remeshing is easy as dynamic simulation can be resumed from previous node locations rather than having to start afresh.

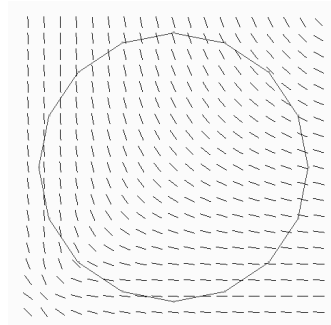
This method is naturally extensible to three dimensions by packing parallelepipeds instead of rectangles to generate anisotropic hexahedral meshes.

Since a large number of inputs is required to use the full capabilities of an anisotropic quadrilateral mesh generator, an adaptation scheme is desirable that can automatically produce the input, in this case – size, aspect ratio, and mesh directions. It is difficult to specify this input in complicated problems without prior knowledge. So a preliminary analysis would present a picture of the loads, material properties, and boundary conditions. Even though it is possible to miss fine features when using a coarse mesh, this limited information, when used for mesh generation, is much better than none at all. When the mesh better reflects the problem being solved, computational cost can be lowered for a desired accuracy, or equivalently for the same cost a better solution can be obtained. The design process is streamlined and cycle times are reduced.

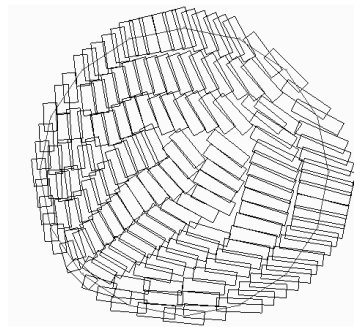
Qualitative methods are used to ascertain how good a mesh is for a specific application. A quantitative approach will give a definite indication of how much one mesh is better than another. This is an important issue that should still be addressed.

## ACKNOWLEDGMENTS

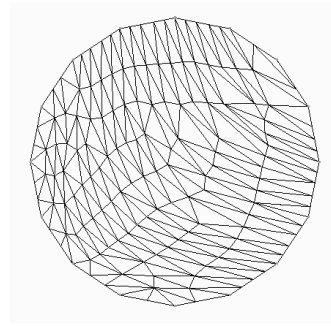
This work is supported in part by a NSF CAREER Award (No. 9985288) and the Pennsylvania Infrastructure Technology Alliance, a partnership of Carnegie Mellon, Lehigh University, and the Commonwealth of Pennsylvania's Department of Economic and Community Development.



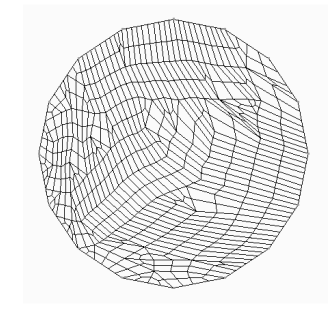
(a) Input mesh directions



(b) Packed rectangles



(c) Delaunay triangulation

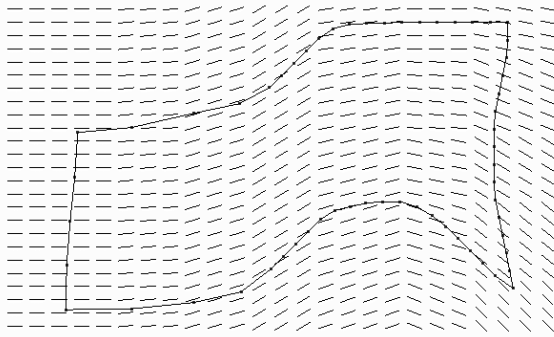


(d) Anisotropic quadrilateral mesh,  
Nodes 621, Elements 564

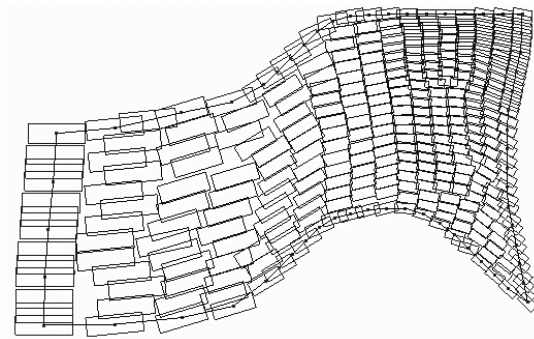
Figure 10 Mesh generation result – 1

## REFERENCES

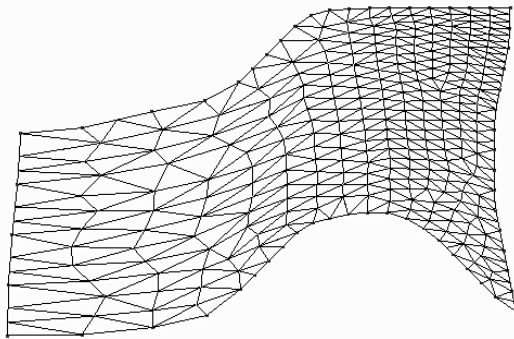
- [1] A. Malanchara and W. Gerstle, "Comparative study of unstructured meshes made of triangles and quadrilaterals," presented at 6 th International Meshing Roundtable., 1997.
- [2] K. Shimada, "Physically-Based Mesh Generation: Automated Triangulation of Surfaces and Volumes via Bubble Packing," in *Mechanical Engineering*. Cambridge, MA, U.S.A.: Massachusetts Institute of Technology, 1993.
- [3] K. Shimada and D. C. Gossard, "Bubble Mesh: Automated triangular meshing of non-manifold geometry by sphere packing," presented at Third Symposium on Solid Modeling and Applications, 1995.
- [4] K. Shimada, A. Yamada, and T. Itoh, "Anisotropic Triangulation of Parametric Surfaces via Close Packing of Ellipsoids," *To appear in International Journal on Computational Geometry and Applications*.
- [5] K. Shimada, J.-H. Liao, and T. Itoh, "Quadrilateral Meshing with Directionality Control through the Packing of Square Cells," presented at 7 th International Meshing Roundtable, 1998.
- [6] K. Ho-Le, "Finite element mesh generation method : A review and classification," *Computer Aided Design*, vol. 20(1), pp. 27-38, 1988.
- [7] N. Sapidis and R. Perucchio, "Advanced techniques for automatic finite element meshing from solid models," *Computer Aided Design*, vol. 8(4), pp. 248-253, 1989.
- [8] M. S. Shephard and e. al., "Trends in automatic three-dimensional mesh generation," *Computers and Structures*, vol. 30(1/2), pp. 421-429, 1988.
- [9] W. C. Thacker, "A brief review of techniques for generating irregular computational grids," *International Journal for Numerical Methods in Engineering*, vol. 15, pp. 1335-1341, 1980.
- [10] S. Owen, "A Survey of Unstructured Mesh Generation Technology," <http://www.andrew.cmu.edu/user/sowen/survey/index.html>.
- [11] A. Hines, "Comparison of finite element meshing software packages," 1999, <http://www.andrew.cmu.edu/~sowen/hines99.html>.
- [12] R. Schneiders, "List of meshing software," <http://www-users.informatik.rwth-aachen.de/~roberts/software.html>.
- [13] E. A. Heighway, "A mesh generator for automatically subdividing irregular polygons into quadrilaterals," *IEEE transactions on Magnetics, Mag-19*, 1983.
- [14] B. P. Johnston, J. M. S. Jr., and A. Kwasnik, "Automatic conversion of triangular finite element meshes to quadrilateral elements," *International Journal for Numerical Methods in Engineering*, vol. 31, 1991.
- [15] K. Shimada, A. Yamada, and T. Itoh, "Automated conversion of 2d triangular meshes into quadrilateral meshes," presented at International Conference on Computational Engineering Science, 1995.
- [16] T. Itoh, K. Shimada, K. Inoue, A. Yamada, and T. Furuhashi, "Automatic conversion of 2D triangular mesh into quadrilateral mesh with directionality control," *presented at 7 th International Meshing Roundtable, 1998*.
- [17] Y. T. Lee, "Automatic finite-element mesh generation," *ACM Transactions on Graphics*, vol. 3, pp. 287-311, 1984.
- [18] Y. T. Lee, "Automatic Finite Element Mesh Generation based on Constructive Solid Geometry," . Leeds, England: University of Leeds, 1983.
- [19] T. D. Blacker and M. B. Stephenson, "Paving: A new approach to automated quadrilateral mesh generation," *International Journal for Numerical Methods in Engineering*, vol. 32, pp. 811-847, 1991.
- [20] S. J. Owen, M. L. Staten, S. A. Canann, and S. Saigal, "Q-Morph: An Indirect Approach to Advancing Front Quad Meshing," *International Journal for Numerical Methods in Engineering*, vol. 9, pp. 1317-1340, 1999.
- [21] M. J. Castro-Diaz, F. Hecht, and B. Mohammadi, "New progress in anisotropic grid adaptation for inviscid and viscous flow simulations," presented at 4th International Meshing Roundtable, 1995.
- [22] H. Borouchaki, P. J. Frey, and P. L. George, "Unstructured triangular-quadrilateral mesh generation application to surface meshing," presented at 5th International Meshing Roundtable, 1996.
- [23] F. J. Bossen and P. S. Heckbert, "A Pliant Method for Anisotropic Mesh Generation," presented at 5th International Meshing Roundtable, 1996.
- [24] X.-Y. Li, S.-H. Teng, and A. Ungor, "Biting Ellipses to Generate Anisotropic Mesh," presented at 8 th International Meshing Roundtable, 1999.
- [25] E. Kreyszig, *Advanced engineering mathematics*, 5 ed: New Age International (P) Limited., 1997.
- [26] D. A. Field, "Laplacian smoothing and delaunay triangulation," *Communications in Applied Numerical Methods*, vol. 4, pp. 709 - 719, 1988.
- [27] S. W. Sloan, "A fast algorithm for constructing Delaunay triangulations in plane," *Advanced Engineering Software*, vol. 9, pp. 34-42, 1987.



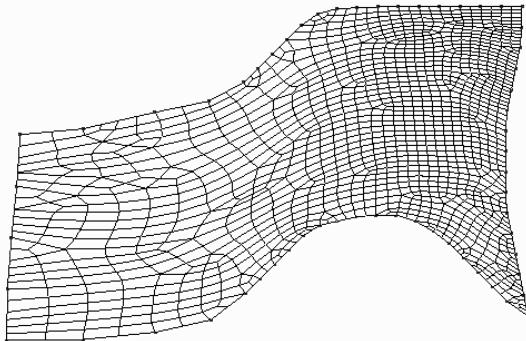
(a) Input mesh directions



(b) Packed rectangles

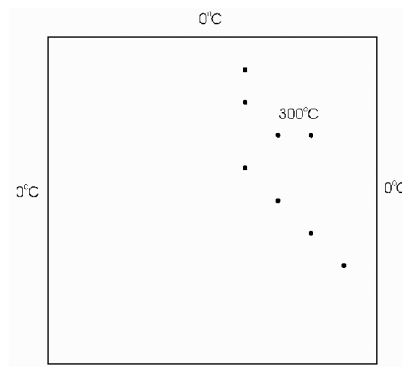


(c) Delaunay triangulation

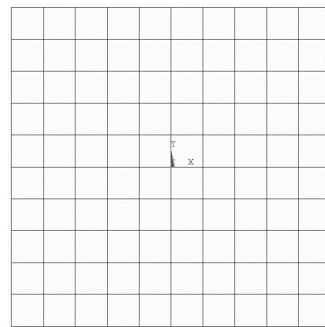


(d) Anisotropic quadrilateral mesh  
Nodes 1303, Elements 1212

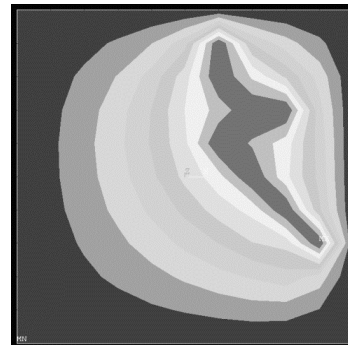
Figure 11 Mesh generation results – 2



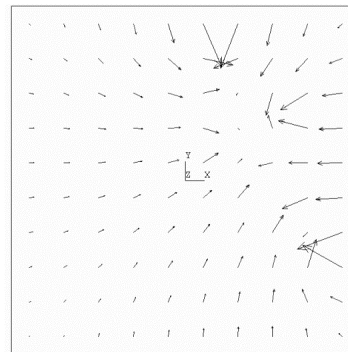
(a) Input geometry and loads



(b) Coarse mesh (121 nodes, 100 elements)

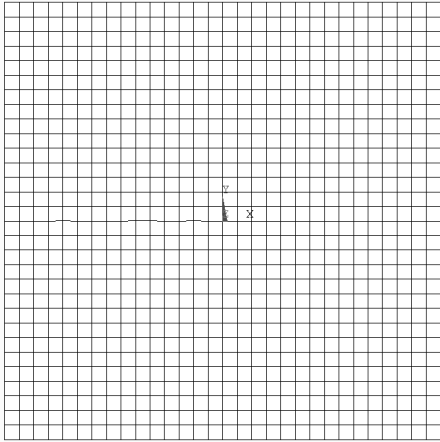


(c) Temperature contours

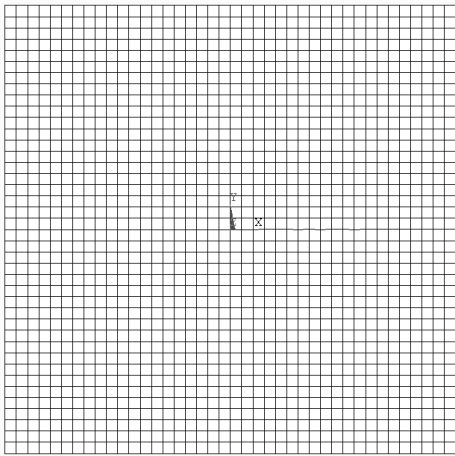


(d) Temperature gradients

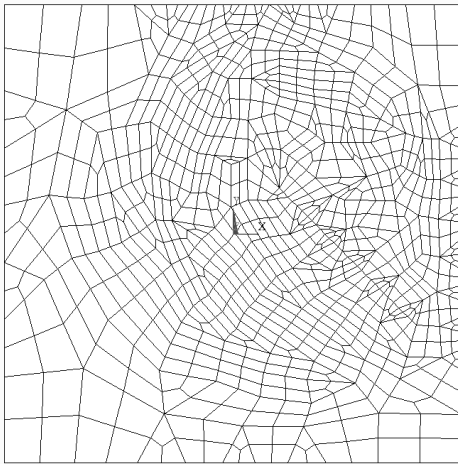
Figure 12 Preliminary analysis using a coarse mesh



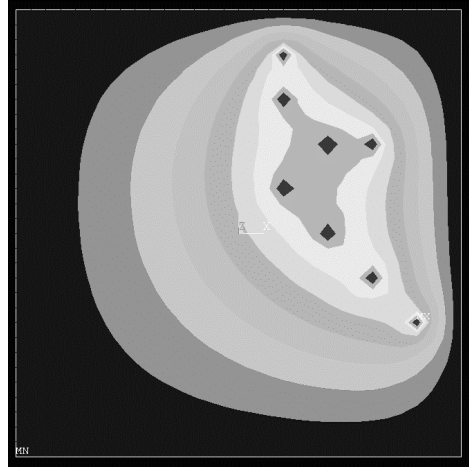
(a) Regular mesh (961 nodes, 900 elements)



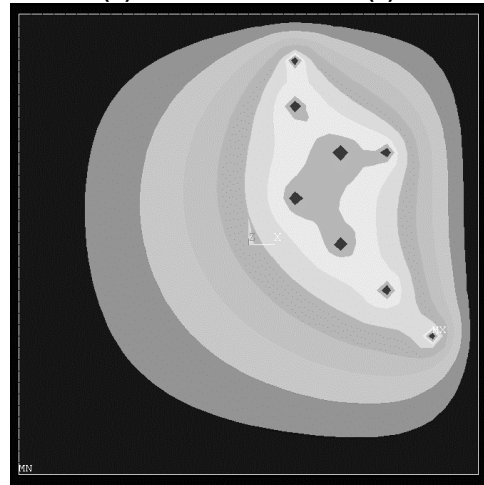
(b) Regular mesh (1681 nodes, 1600 elements)



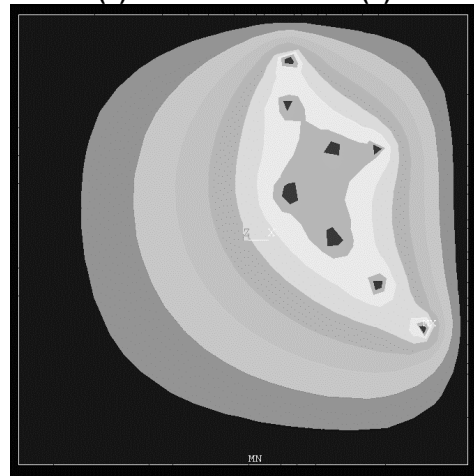
(c) Generated mesh (879 nodes, 828 elements)



(d) Solution with mesh (a)



(e) Solution with mesh (b)



(f) Solution with mesh (c)

Figure 13 Meshes and temperature contours – Steady-state heat-transfer