

# HIGH QUALITY ANISOTROPIC TETRAHEDRAL MESH GENERATION VIA ELLIPSOIDAL BUBBLE PACKING

Soji Yamakawa<sup>1</sup> and Kenji Shimada<sup>2\*</sup>

<sup>1</sup>*Carnegie Mellon University, Pittsburgh, PA, U.S.A. soji@andrew.cmu.edu*

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

## ABSTRACT

This paper presents a new computational method for anisotropic tetrahedral meshing that (1) can control shapes of the elements by an arbitrary anisotropy function, and (2) can avoid ill-shaped elements induced from poorly distributed node locations. Our method creates a tetrahedral mesh in two steps. First our method obtains node locations through a physically based particle simulation, which we call 'bubble packing.' Ellipsoidal bubbles are closely packed on the boundary and inside a geometric domain, and nodes are placed at the centers of the bubbles. Our method then connects the nodes to create a tet mesh by the advancing front method. Experimental results show that our method can create a high quality anisotropic tetrahedral mesh that conforms well to the input anisotropy.

**Keywords:** tetrahedral mesh, bubble packing, anisotropy, graded, unstructured

## 1. INTRODUCTION

This paper presents a new computational method for anisotropic tetrahedral meshing. Our method has two major advantages: (1) our method can control shapes of elements by an arbitrary anisotropy function, and (2) our method can avoid ill-shaped elements induced from poorly distributed node locations. Tetrahedral mesh is widely used in commercial applications such as finite element packages and computer aided geometric design systems. Although many algorithms are available for tetrahedral meshing, such as the Delaunay method [6,7,15] and the advancing front method [4,10] most of them are applicable only to isotropic graded meshes.

In our method, an anisotropy controls the shape of the mesh over the domain. Typically, the element shapes are controlled by a prescribed anisotropy function

$\mathbf{M} = \mathbf{M}(x, y, z)$ , which defines the anisotropy at point  $(x, y, z)$ . The orientation of the elements must be aligned with the given anisotropy. And the shapes of the elements must be 'stretched' by the given aspect ratio. For example, in CFD applications, the gradient of the solution in the flow direction is generally smaller than the gradient in the orthogonal direction. In such cases, the density of the elements must be higher in the large gradient direction and lower in the small gradient direction. In other words, elements must be aligned and stretched in the flow direction.

Although the demand for the anisotropic tetrahedral meshes is high, most previous methods of anisotropic meshing are for either two-dimensional domains or surface domains. A few anisotropic tetrahedral meshing techniques have been developed, but they are application specific, and cannot deal with an arbitrary anisotropy function. We will review the previous work in Section 2.

---

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

Among those previously presented methods, the Bubble Mesh method has showed good results for both isotropic and anisotropic triangular and quadrilateral meshing [11,12,13,14]. The idea of the Bubble Mesh method came from the behavior of soap bubbles in nature. If we pack soap bubbles in a volumetric domain, the bubbles will naturally form a hexagonal pattern. The centers of the bubbles yield ideal node locations that, when connected, generate a high quality isotropic tetrahedral mesh.

In this paper, we extend the Bubble Mesh method to anisotropic tetrahedral meshing by packing ellipsoids instead of spheres. Our method takes as inputs a geometric domain  $\Omega$  and anisotropy  $\mathbf{M}(x,y,z)$  and creates an anisotropic tetrahedral mesh  $A$ . The meshing process consists of two major steps:

- (1) Creating node locations by bubble packing, detailed in Section 4
- (2) Connecting nodes considering the anisotropy by the advancing front method, detailed in Section 5

The pattern of the packed ellipsoids yields node locations suitable for anisotropic tetrahedral meshing because it mimics ideal Voronoi polyhedra of an anisotropic tetrahedral mesh. The method creates ideally spaced nodes with respect to the given anisotropy and thus can avoid ill-shaped elements that result when two nodes are too close to each other or where the density of the nodes is less than ideal. Our experimental results show that the proposed method creates a quality anisotropic mesh that conforms well to a specified anisotropy.

In the following, we first review related work in Section 2. Then we explain the details of the bubble packing and mesh generation process in Sections 4 and 5, followed by experimental results in Section 6 and conclusions.

## 2. PREVIOUS WORK

Most anisotropic meshing techniques pertain to 2D and surface triangular meshing. A few anisotropic tetrahedral meshing algorithms have been presented, but they are application-specific and cannot deal with an arbitrary anisotropy. We review the development of anisotropic meshing techniques in the rest of this section.

Castro-Diaz et al. propose a technique to adapt a triangular mesh to an appropriate anisotropy for CFD applications [3]. The method takes an initial triangular mesh and computes a CFD solution. An anisotropy is then computed based on the results of the initial CFD solution. The mesh is then adapted to the computed anisotropy. This procedure is repeated until the estimated interpolation error in the solution becomes less than a prescribed threshold.

Bossen and Heckbert introduce a more general anisotropic triangular mesh generation technique [2]. Their method first creates a constrained Delaunay mesh of the target domain. After creating an initial mesh, nodes are adaptively inserted or deleted. Then the mesh is smoothed, refined or retriangulated considering the given anisotropy. Repeating this process will adapt the mesh to the given

anisotropy. They present a modified Delaunay criterion to connect the nodes in an anisotropic fashion. Since a 2x2 matrix transforms a circle into an ellipse, the modified Delaunay criterion uses a circum-ellipse test instead of a circumcircle test.

Borouchaki et al. propose a method for creating anisotropic triangular and quadrilateral meshes [1]. They generalize the Delaunay kernel method [15] for anisotropic triangular meshing. Their method also converts a triangular mesh into quadrilateral mesh.

Shimada et al. introduce a technique that creates an anisotropic triangular mesh on a curved surface [13]. The method packs ellipsoidal bubbles on a curved surface to obtain node locations. Those nodes are connected using a modified Delaunay triangulation. Because this method creates good node locations, virtually no ill-shaped elements are created.

Garimella and Shephard present a method that creates a boundary-aligned anisotropic tetrahedral mesh for CFD applications [5]. Their method first creates elements of the boundary layers through the advancing front method, and then creates elements for the remainder in the domain by another isotropic tetrahedrization technique. This method can control the density of the elements in the direction perpendicular to the boundary by adjusting the speed of the advancing front. However, it is unable to control the anisotropy of the interior elements.

Li et al. introduce a node-placing algorithm called ‘Biting’ for generating an anisotropic triangular mesh [9]. Their method progressively creates node locations from the boundary to the interior of a domain. The method first places nodes on the vertices of the domain  $\Omega$ . When a node is placed, an elliptic region  $\Phi_i$  around the node is removed from the domain  $\Omega$ . Then, new nodes are created at the intersections between  $\Phi_i$  and  $\Omega$ . Repeating this procedure until the entire domain is covered will give quality node locations. In this paper [9], however, no meshed result or quality measurement is provided.

## 3. REPRESENTATION OF ANISOTROPY AND GRADING OF ELEMENT SIZE

In this paper we use the standard tensor-based metric for representing mesh anisotropy. In a 2D coordinate system, a 2x2 matrix represents the anisotropy and element size [1,2,3]. Similarly, a 3x3 matrix represents the anisotropy and element size in a 3D coordinate system [13].

An anisotropy is defined by three orthogonal principal directions and an aspect ratio in each direction. The three principal directions are represented by three unit vectors  $\mathbf{u}$ ,  $\mathbf{v}$ , and  $\mathbf{w}$ , and in these directions the amounts of stretching of a mesh element are represented by three scalar values  $\lambda_u$ ,  $\lambda_v$  and  $\lambda_w$  respectively. Using  $(\mathbf{u}, \mathbf{v}, \mathbf{w})$  and  $(\lambda_u, \lambda_v, \lambda_w)$  we define two matrices  $\mathbf{R}$  and  $\mathbf{S}$

$$\mathbf{R} = \begin{pmatrix} u_x & v_x & w_x \\ u_y & v_y & w_y \\ u_z & v_z & w_z \end{pmatrix}, \mathbf{S} = \begin{pmatrix} \lambda_u & 0 & 0 \\ 0 & \lambda_v & 0 \\ 0 & 0 & \lambda_w \end{pmatrix}$$

By combining matrices  $\mathbf{R}$  and  $\mathbf{S}$ , we obtain a 3x3 positive definite matrix  $\mathbf{M}$  that describes the three-dimensional anisotropy:

$$\mathbf{M} = \mathbf{R}\mathbf{S}\mathbf{R}^T$$

In practice, because an anisotropy varies over the domain, matrix  $\mathbf{M}$  is given as a function of position in three-dimensional space:

$$\mathbf{M} = \mathbf{M}(x, y, z)$$

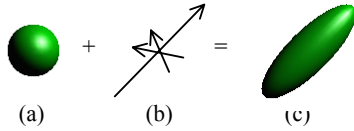
Since  $\mathbf{M}$  is a positive definite matrix,  $\mathbf{M}$  is decomposed into two matrices  $\mathbf{Q}^T$  and  $\mathbf{Q}$  as:

$$\mathbf{M} = \mathbf{Q}^T\mathbf{Q}$$

where

$$\mathbf{Q} = \begin{pmatrix} \sqrt{\lambda_u} & 0 & 0 \\ 0 & \sqrt{\lambda_v} & 0 \\ 0 & 0 & \sqrt{\lambda_w} \end{pmatrix} \mathbf{R}^T$$

Figure 1 illustrates the transformation defined by  $\mathbf{Q}$ . A unit sphere shown in Figure 1 (a) is transformed into an ellipsoid shown in Figure 1 (c) by the inverse of the matrix,  $\mathbf{Q}^{-1}$ , where  $\mathbf{M} = \mathbf{Q}^T\mathbf{Q}$  represents the anisotropy shown in Figure 1 (b).



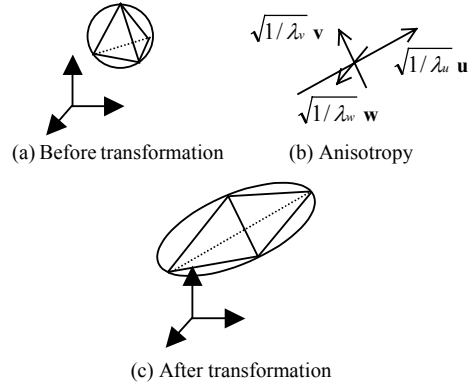
**Figure 1 Anisotropic bubble**

This geometric interpretation implies that we should pack ellipsoids instead of spheres to obtain good node locations for anisotropic meshing. Equally spaced node locations are best for an isotropic tetrahedral mesh. Because a sphere defines a surface equidistant from its center, we can obtain good node locations for an isotropic tetrahedral mesh by closely packing spheres. Similarly, Figure 1 suggests that an ellipsoid defines an equi-distance surface which takes an anisotropy into account. In other words, if we closely pack ellipsoids, the centers of the ellipsoids will be equally spaced with respect to the anisotropy.

Figure 2 shows an example of the transformation. Figure 2 (a) is an ideal unit tetrahedron without anisotropy. Here, the anisotropy is specified by three unit vectors,  $\mathbf{u}$ ,  $\mathbf{v}$ , and  $\mathbf{w}$ , and aspect ratios are defined as  $\sqrt{1/\lambda_u} = 3.0$ ,  $\sqrt{1/\lambda_v} = 1.0$ ,  $\sqrt{1/\lambda_w} = 1.0$ . The arrows in Figure 2 (b) show the three vectors  $\sqrt{1/\lambda_u} \mathbf{u}$ ,  $\sqrt{1/\lambda_v} \mathbf{v}$  and

$\sqrt{1/\lambda_w} \mathbf{w}$ . If the corresponding 3x3 matrix is  $\mathbf{M} = \mathbf{Q}^T\mathbf{Q}$ , the matrix  $\mathbf{Q}^{-1}$  transforms the unit tetrahedron shown in Figure 2 (a) into the tetrahedron shown in Figure 2 (c). With this transformation, the original tetrahedron is stretched in the  $\mathbf{u}$  direction by a factor of 3.0. As a result, the tetrahedron in Figure 2 (c) is an ideal tetrahedron for the specified anisotropy.

Recall that the anisotropy matrix  $\mathbf{M}$ , was composed in part of a matrix of  $\lambda_u$ ,  $\lambda_v$  and  $\lambda_w$ . Since  $(\lambda_u, \lambda_v, \lambda_w)$  represents an element size, the 3x3 matrix  $\mathbf{M}$  describes both an element's anisotropy and its size. Thus, in the following section, if we use the term 'anisotropy,' it includes the grading of element size. Since the matrix  $\mathbf{M}$  is equivalent to anisotropy, we will simply say 'anisotropy  $\mathbf{M}$ ' to refer to the anisotropy represented by matrix  $\mathbf{M}$ .



**Figure 2 An ideal element with anisotropy**

#### 4. ANISOTROPIC NODE DISTRIBUTION BY PACKING ELLIPSOIDS

This section describes one of the two steps of our meshing procedure. The main technical issue of the first step is a node-locating algorithm that we call 'bubble packing.'

##### 4.1 Outline of Bubble Packing

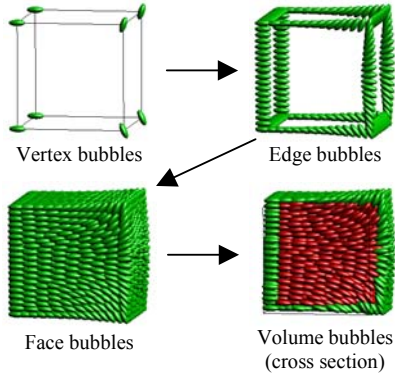
The bubble packing process consists of four sub-steps, as illustrated in Figure 3.

1. Place bubbles on the vertices of the target domain  $\Omega$
2. Pack bubbles on the edges of the target domain  $\Omega$
3. Pack bubbles on the faces of the target domain  $\Omega$
4. Pack bubbles in the interior of the target domain  $\Omega$

Note that we are packing ellipsoidal bubbles that conform to the given anisotropy.

During sub-processes 2 through 4, bubbles are created in the target domain  $\Omega$  and moved to stable positions by

physically-based particle simulation. To obtain well-distributed node locations, the bubbles must be closely packed inside the domain. Closely packed bubbles will minimize gaps and overlaps. If we pack an appropriate number of bubbles closely in the domain with respect to anisotropy  $\mathbf{M}$ , the bubbles will move to a stable configuration in which all the proximity-based inter-bubble forces are balanced, as described in the next section.



**Figure 3 Packing ellipsoidal bubbles in a domain**

#### 4.2 Computation of the Motion of the Bubbles

To run the simulation, we have to derive the equation of motion that governs the dynamic behavior of the bubbles. Once we know the equation of motion, we can apply a standard numerical integration scheme such as Euler's method or the 4th order Runge-Kutta method to simulate the motion of the bubbles. To derive the equation of motion, we first formulate the forces acting on the bubbles. Then, we derive the second order differential equation that governs the motion of the bubbles by adding a point mass for each bubble and the effect of viscous damping.

In our method, forces acting on the bubbles are computed approximately by a mass-spring-damper model. In this model, two kinds of forces act on the bubbles. One is an inter-bubble force that is analogous to a non-linear spring. The inter-bubble force is a proximity force and acts only between two adjacent bubbles. The other type of force is due to viscous damping.

In the mass-spring-damper model, each bubble has two state variables, position and velocity, and two attributes, mass and a damping coefficient. We denote the state variables of the  $i$ th bubble as:

$\mathbf{x}_i$  : Position of bubble  $i$

$\dot{\mathbf{x}}_i$  : Velocity of bubble  $i$

We assume that all bubbles have the same mass  $m$  and the same damping coefficient  $c$ , that is:

$$m_i = m$$

$$c_i = c$$

Having these state variables and attributes, we are ready to formulate the forces acting on the bubbles.

The inter-bubble force is approximated as a force produced by a non-linear spring connecting the centers of adjacent bubbles. Figure 4 shows an example of this spring. In order to compute the inter-bubble force between bubble  $i$  and bubble  $j$ , we need to know the current length and the neutral length of the spring. The current length  $l$  of the spring between bubble  $i$  and bubble  $j$  is computed as:

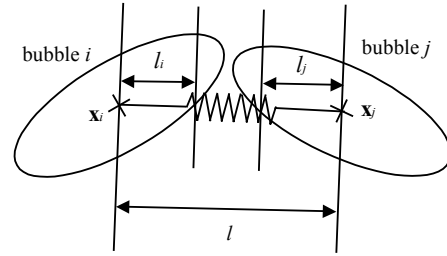
$$l = \|\mathbf{x}_i - \mathbf{x}_j\|$$

where  $\mathbf{x}_i$  is the center of bubble  $i$ ,  $\mathbf{x}_j$  is the center of bubble  $j$ , and  $\|\cdot\|$  denotes the  $L_2$  Euclidean norm.

We define the neutral length of the spring  $l_0$  as:

$$l_0 = l_i + l_j$$

where  $l_i$  is the distance from point  $\mathbf{x}_i$  to the intersection between: (1) the line segment connecting  $\mathbf{x}_i$  and  $\mathbf{x}_j$ , and (2) the boundary of bubble  $i$ .  $l_j$  is the distance from point  $\mathbf{x}_j$  to the intersection between the same line segment and the boundary of bubble  $j$ .



**Figure 4 Non-linear spring**

After finding  $l$  and  $l_0$ , the inter-bubble force is computed as a function of the ratio of  $l$  and  $l_0$ . Two adjacent bubbles either attract each other, repel each other or do not interact depending on the value of  $l/l_0$ . Figure 5 shows how two bubbles interact: (1) when  $l/l_0 = 1$ , the bubbles are at a stable distance and do not either attract or repel each other (Figure 5 (a)), (2) when  $l/l_0 < 1$ , they repel each other (Figure 5 (b)), (3) when  $1 < l/l_0 < 1.5$ , they attract each other (Figure 5 (c)), and (4) when  $1.5 < l/l_0$ , again they do not interact (Figure 5 (d)).

As in the previous Bubble Mesh methods, we define the force of the non-linear spring using a cubic function:

$$f(w) = \begin{cases} k(1.25w^3 - 2.375w^2 + 1.125) & \text{for } 0.0 \leq w \leq 1.5 \\ 0 & \text{otherwise} \end{cases}$$

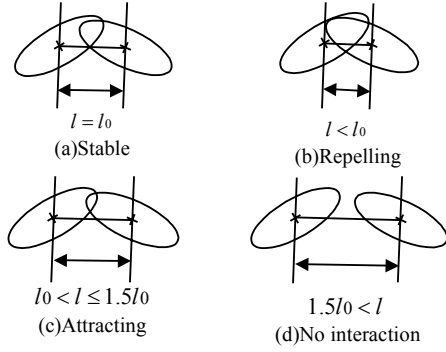
where  $w$  is the ratio of  $l$  and  $l_0$ , i.e.  $w = l/l_0$ , and  $k$  is a constant.

Since multiple bubbles can be adjacent to a given bubble  $i$ , the total inter-bubble force acting on the  $i$ th bubble can be computed by:

$$\mathbf{f}_i = \sum_j f_{ij} \frac{\mathbf{x}_i - \mathbf{x}_j}{\|\mathbf{x}_i - \mathbf{x}_j\|}$$

where

$$f_{ij} = \begin{cases} \text{force applied to bubble } i \text{ by bubble } j & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases}$$



**Figure 5 Schematic of proximity force**

In order to make a system of bubbles converge to a stable configuration, damping must be added to the system. We define a damping force acting on bubble  $i$  to be proportional to the velocity of the bubble:

$$-c\dot{\mathbf{x}}_i$$

Finally, we can write down the second order ordinary differential equation that governs the motion of the bubbles as:

$$m\ddot{\mathbf{x}}_i + c\dot{\mathbf{x}}_i = \mathbf{f}_i$$

Numerical integration such as Euler's method or the 4th order Runge-Kutta method can be used to solve this ordinary differential equation.

### 4.3 Population Control

Another important procedure during bubble packing is a process called 'adaptive population control.' Its purpose is to adjust the number of bubbles so that there are no significant gaps or overlaps in the final packed configuration. As mentioned before, bubbles should be closely packed in order to obtain well distributed node locations. If the number of bubbles is too small, large gaps exist between bubbles. In such a case, the bubbles may form a random pattern, yielding a low quality mesh. On the other hand, if the number of bubbles is too large, the bubbles tend to form an orthogonal pattern, which is not suitable for tetrahedral meshing. To obtain the well distributed node locations, it is important to adjust the number of bubbles during dynamic simulation.

The optimal number of bubbles is a function of the domain geometry and the specified anisotropy, that is:

$$N_{opt} = N(\Omega, \mathbf{M})$$

where

$$\begin{aligned} N_{opt} &: \text{Appropriate number of bubbles} \\ \Omega &: \text{Target domain} \\ \mathbf{M} &: \text{Anisotropy} \end{aligned}$$

If the domain geometry is simple (i.e., no sharp corners or small features) and if a constant anisotropy is given, an appropriate number of bubbles can be estimated by:

$$N = \alpha \frac{V(\Omega)}{V_b}$$

where

$$\begin{aligned} V(\Omega) &: \text{Volume of the target domain} \\ V_b &: \text{Volume of a bubble} \\ \alpha &: \text{Constant} \end{aligned}$$

We can use this formula to estimate an appropriate number of bubbles in a small volume around a sample point. Although our program makes use of the formula to make an initial guess of the bubble configuration, when the domain is complicated and a varying anisotropy is specified, this formula gives only a rough estimate of an appropriate number of bubbles.

Because initial guess cannot be accurate, we need to control the number of bubbles adaptively through the packing process. We take advantage of the computation of the inter-bubble force to find over-populated and under-populated regions. If a bubble is in an over-populated region, it receives repelling forces. In contrast, if a bubble is in an under-populated region, the bubble receives more attracting forces than repelling forces. Our program deletes bubbles from over-populated regions and adds bubbles in under-populated regions once every a few iterations. Appropriate thresholds must be chosen to identify over-/under-populated regions.

## 5. FINDING NODE CONNECTIVITY CONSIDERING ANISOTROPY

In the second step of our meshing procedure, we connect nodes to construct an anisotropic tetrahedral mesh. Although many tetrahedrizations are possible for a given set of nodes, there are only a limited number of tetrahedrizations that represent high quality mesh elements with respect to the given anisotropy. Thus, it is important that our method chooses an appropriate tetrahedrization.

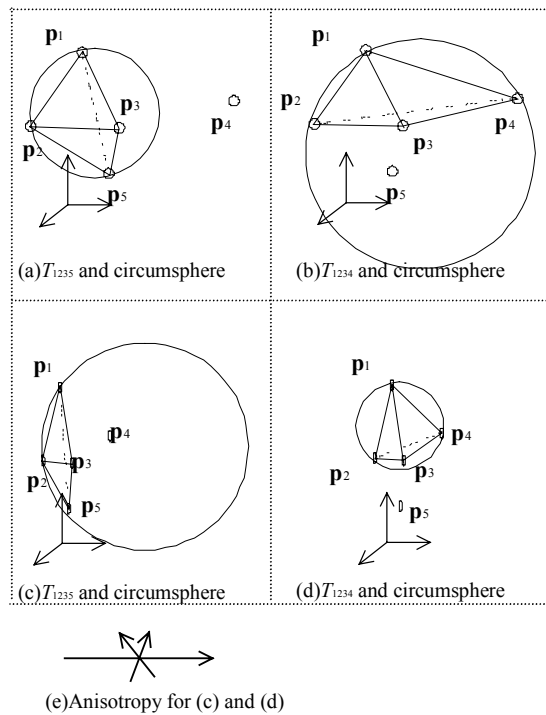
### 5.1 Modified Delaunay Criterion

During the generation of an anisotropic tetrahedral mesh, it is necessary to align elements according to a specified anisotropy. We use a modified Delaunay criterion to test the alignment of elements as described below.

The original Delaunay criterion is based on a 'circumsphere test.' If no nodes are inside the circumscribed sphere of a tetrahedron  $T$ ,  $T$  is said to be valid with respect to the circumsphere test. The modified Delaunay criterion is also

based on the circumsphere test; however, the modified Delaunay criterion applies the circumsphere test after transforming coordinates by matrix  $\mathbf{Q}$ , where  $\mathbf{Q}^T\mathbf{Q}$  is the decomposition of the given anisotropy  $\mathbf{M}$ .

Figure 6 shows a comparison between the original Delaunay criterion and the modified Delaunay criterion. There are 5 nodes  $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4$  and  $\mathbf{p}_5$ . We denote a tetrahedron consisting of  $\mathbf{p}_a, \mathbf{p}_b, \mathbf{p}_c$  and  $\mathbf{p}_d$  as  $T_{abcd}$ . Two tetrahedra  $T_{1235}$  and  $T_{1234}$  are tested by the original Delaunay criterion in Figures 6 (a) and 6 (b), and by the modified Delaunay criterion in Figures 6 (c) and 6 (d). Note that the coordinates in Figure 6 (c) and Figure 6 (d) are transformed by matrix  $\mathbf{Q}$ , where  $\mathbf{Q}^T\mathbf{Q}$  is the decomposition of the given anisotropy  $\mathbf{M}$  shown in Figure 6 (e).



**Figure 6 Modified Delaunay criterion**

Note that  $\mathbf{p}_4$  is exterior to the circumsphere of  $T_{1235}$  in Figure 6 (a) and  $\mathbf{p}_5$  is interior to the circumsphere of  $T_{1234}$  in Figure 6 (b). As a result, according to the original Delaunay criterion,  $T_{1235}$  is valid and  $T_{1234}$  is invalid. In contrast, if we run the circumsphere test after transforming coordinates by the given anisotropy, the test gives a different result. As shown in Figure 6 (c),  $\mathbf{p}_4$  is now interior to the circumsphere of  $T_{1235}$ , and as shown in Figure 6 (d),  $\mathbf{p}_5$  is exterior to the circumsphere of  $T_{1234}$ .

Consequently, according to the modified Delaunay criterion,  $T_{1235}$  is invalid and  $T_{1234}$  is valid.

It can be observed that  $T_{1234}$  has a better shape than  $T_{1235}$  in the transformed coordinate system, or in other words,  $T_{1234}$  is aligned to the anisotropy in the untransformed coordinate system. As can be seen in this example, the modified Delaunay criterion chooses a combination of tetrahedra that align well to the given anisotropy.

## 5.2 Advancing Front

We employ the advancing front method to connect nodes. Now that we know the way to test the alignment of a tetrahedron, what we need next is a node-connecting algorithm to create a tetrahedral mesh.

Our algorithm is similar to the algorithm presented by Fleischmann and Selberherr [4], which connects pre-created nodes and boundary triangles to form tetrahedral elements. Their method takes advantage of two facts: (1) Delaunay tetrahedrization exists for any set of nodes, and (2) a sphere does not have an orientation. However, we do not know if all elements can satisfy the Delaunay criterion in an anisotropic mesh. Furthermore, the orientation and aspect ratio of a circum-ellipsoid changes for different tetrahedra. Thus, our method more thoroughly checks the validity of elements during the procedure.

As shown in Figure 7 (a), we first mesh boundaries into a triangular mesh considering anisotropy. We use a modified Delaunay triangulation to create a complete triangular mesh [1,2,13]. This triangular mesh is the so called ‘front.’ The following notations will be used throughout the remainder of this section:

$F$	Current front
$V(F)$	The volume enclosed by the current front $F$
$P$	A triangle and node pair
$T(P)$	A tetrahedron formed by a triangle-node pair $P$
$V(T(P))$	The volume defined by a tetrahedron $T(P)$

After creating a triangular mesh  $F$ , tetrahedral elements are created one by one. A triangle is connected to a node to create a tetrahedral element. Unlike the original advancing front method, no new nodes are created during this process. Instead, our method searches out triangle-node pairs  $P$  that can be removed from the current volume  $V(F)$ .  $P$  is said to be removable if  $T(P)$  does not intersect with the current front  $F$ , and  $V(T(P))$  is inside  $V(F)$ . When  $P$  is removed,  $T(P)$  is added to the element list, and  $F$  is updated so that  $V(T(P))$  is removed from  $V(F)$ . In other words,  $V(T(P))$  is meshed.

In our method, the program searches for a pair  $P$  such that  $T(P)$  satisfies the modified Delaunay criterion. If an

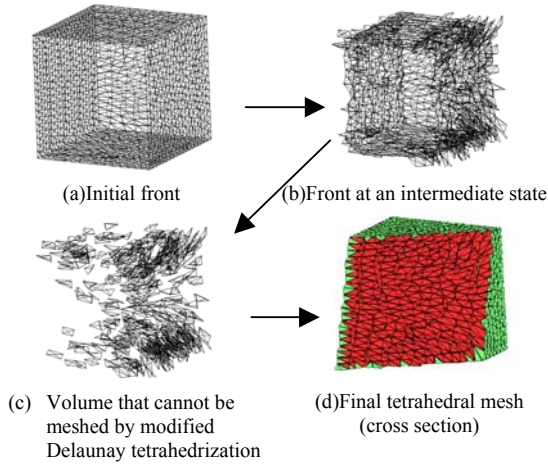
appropriate pair  $P$  is found,  $V(T(P))$  is meshed. Figure 7 (b) shows the front after a portion of the volume is meshed.

There is no guarantee, however, that all the elements will satisfy the modified Delaunay criterion. As a result, some volume will remain unmeshed, as shown in Figure 7 (c). In such a case, the program searches among all possible pairs for the one that yields the best quality element, regardless of the Delaunay criterion. If the pair  $P$  is found,  $V(T(P))$  is meshed. This process is repeated until all removable pairs are meshed.

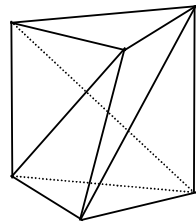
Unfortunately, some sub-volumes can remain unmeshed at the end, typically when the remaining volume forms a shape known as a twisted prism (Figure 8). In such cases, we need extra procedures to mesh the remaining volume. Currently, if some volume remains unmeshed, our method tries the following procedures:

- Undo elements adjacent to the unmeshed volume (i.e., augment the current volume  $V(F)$  by the adjacent elements) and try a different tetrahedrization.
- Run some more iterations of bubble packing and mesh the domain again.

Finally, the domain is meshed into an anisotropic tetrahedral mesh as shown in Figure 7 (d).



**Figure 7 Node connection by an advancing front and modified Delaunay criterion**



**Figure 8 Twisted prism**

### 5.3 Quality Improvement by Edge Swapping

The final procedure of the meshing process further improves the quality by edge swapping. There are two reasons why this process is needed:

- Similar to the original Delaunay criterion, the modified Delaunay criterion cannot eliminate slivers.
- Poorly shaped elements may be created where the front meets the opposing front.

We employed Joe's local transformation, or edge swapping method [8]. This technique improves the mesh quality by modifying the topology of the mesh. The technique is based on the fact that there are a limited number of possible tetrahedrization schemes for five nodes. If a current tetrahedrization  $A_1$  for five nodes includes the worst element  $T_{worst}$  and if another valid tetrahedrization  $A_2$  for the five nodes has no worse element than  $T_{worst}$ , the tetrahedrization  $A_2$  is swapped for  $A_1$ . We also perform multi-step swapping operations to eliminate ill-shaped elements efficiently.

Edge swapping is highly effective at eliminating ill-shaped elements. Typically, all elements that have minimum dihedral angles of less than 10 degrees are eliminated for a convex domain. Detailed results are presented in the next section.

## 6. RESULTS AND QUALITY MEASUREMENT

In this section, we show some experimental results. Quality measurements are also presented for each result. We chose the minimum dihedral angle as a quality measurement. The dihedral angles of an equilateral tetrahedron are approximately 72 degrees. If a tetrahedron  $T$  becomes flatter (i.e., the quality lessens), the minimum dihedral angle of  $T$  becomes smaller.

In anisotropic meshing, the dihedral angles should be computed taking anisotropy into account. If a tetrahedron  $T$  consists of vertices  $\mathbf{p}_1$ ,  $\mathbf{p}_2$ ,  $\mathbf{p}_3$  and  $\mathbf{p}_4$ , we compute the minimum dihedral angle of  $T$  as follows:

- (1) Compute an average anisotropy  $\mathbf{M}_{avg}$  as:

$$\mathbf{M}_{avg} = \mathbf{M} \left( \frac{\mathbf{p}_1 + \mathbf{p}_2 + \mathbf{p}_3 + \mathbf{p}_4}{4} \right)$$

- (2) Transform  $\mathbf{p}_1$ ,  $\mathbf{p}_2$ ,  $\mathbf{p}_3$  and  $\mathbf{p}_4$  as:

$$\mathbf{p}'_n = \mathbf{Q}\mathbf{p}_n, \quad n = 1, 2, 3, 4$$

where

$$\mathbf{M}_{avg} = \mathbf{Q}^T \mathbf{Q}$$

- (3) Compute the minimum dihedral angle using  $\mathbf{p}'_1$ ,  $\mathbf{p}'_2$ ,  $\mathbf{p}'_3$  and  $\mathbf{p}'_4$

Mesh quality is measured by both the worst element quality and the overall quality. The worst minimum dihedral angle of all elements measures the quality of the worst element,

while the histogram of minimum dihedral angles shows the overall quality.

Figures 10, 11 and 12 show some results of our method and quality measurements for each. For each figure, bubble packing, tetrahedral mesh, anisotropy, statistics and quality measurement are presented. The graphs of the quality measurements show histograms of the minimum dihedral angles in each case (degree vs. number of elements).

Figure 10 shows the meshed result of a cylinder. In this example, the cylinder is meshed into equally sized isotropic elements. The minimum dihedral angle of all elements is 14.2 degrees and the histogram of minimum dihedral angles has a peak around 40 degrees.

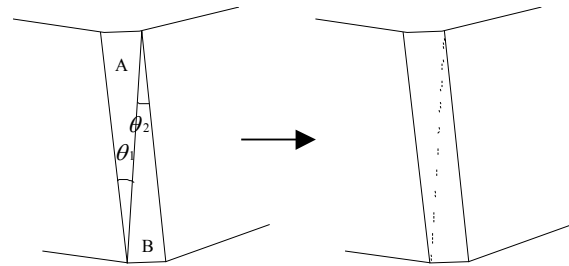
Figure 11 shows the meshed result of a mechanical part that has a cylindrical feature on the base. Anisotropy is given so that the elements are aligned to the cylindrical feature. We also define the anisotropy so that the aspect ratio gradually changes from 1:1 at the centroid to 1:3 at the outside wall of the cylindrical feature. The minimum dihedral angle of all elements is 8.0 degrees and the histogram of minimum dihedral angles again has a peak around 40 degrees.

Figure 12 shows the meshed result of a dinosaur. In this example, elements are aligned with the surface of the dinosaur, and small elements are created in small features such as the legs, arms and head. The minimum dihedral angle of all elements is 5.5 degrees. The histogram of minimum dihedral angles has a peak around 40 degrees.

The cylinder case shows high overall quality and high quality of the worst element. The mechanical part case and the dinosaur case show high overall quality, but the quality of the worst element is relatively low compared to the cylinder case. The worst elements of the mechanical part and the dinosaur were created where polygons of the original domain had sharp corners. Currently, we cannot avoid small dihedral angles if the original geometry has sharp corners. In such cases, we could eliminate most sharp corners by grouping polygons as shown in Figure 9. In Figure 9, triangle A and triangle B have sharp corners,  $\theta_1$  and  $\theta_2$ . If we combine polygon A and polygon B into one group, we can avoid  $\theta_1$  and  $\theta_2$ .

As for time complexity, we found that anisotropic cases require more computational time than isotropic cases. Although the mechanical part has almost the same number of nodes and elements as the cylinder, the mechanical part took 38% longer time than the cylinder to mesh.

For the cylinder case, it took 900 iterations of the bubble packing until the bubbles reached stability on a Pentium III 500MHz PC with 128 MB of memory. The total bubble packing procedure took 125 seconds. Then it took another 36 seconds to connect the nodes. For the mechanical part, it took 1250 iterations of bubble packing until the bubbles reached stability. The total bubble packing procedure took 160 seconds. Then it took another 64 seconds to connect the nodes.



**Figure 9 Grouping polygons to avoid small angles**

We are currently implementing more efficient proximity check algorithms used in bubble packing and the advancing front method to reduce computational time significantly.

## 7. CONCLUSION

We have presented a method for creating a high quality anisotropic tetrahedral mesh. We showed that a 3x3 matrix can completely represent an anisotropy and grading of the element size. We also showed that the 3x3 matrix can be interpreted geometrically as a transformation from a sphere to an ellipsoid.

Our method first creates nodes using bubble packing. Bubbles are packed on the boundary of and inside the target geometric domain. Then the bubbles are moved to a stable configuration iteratively by a physically based particle simulation. Because the anisotropy is represented by a 3x3 matrix, which transforms a sphere into an ellipsoid, we use ellipsoidal bubbles to comply with the given anisotropy.

After creating nodes, the nodes are connected by a modified advancing front method. Unlike the original advancing front method, we do not create new nodes during the process. Our method searches for a triangle-node pair that makes a high quality element. Repeating this procedure until the front disappears, and applying edge swapping as a post process, we can obtain a well-shaped tetrahedral mesh.

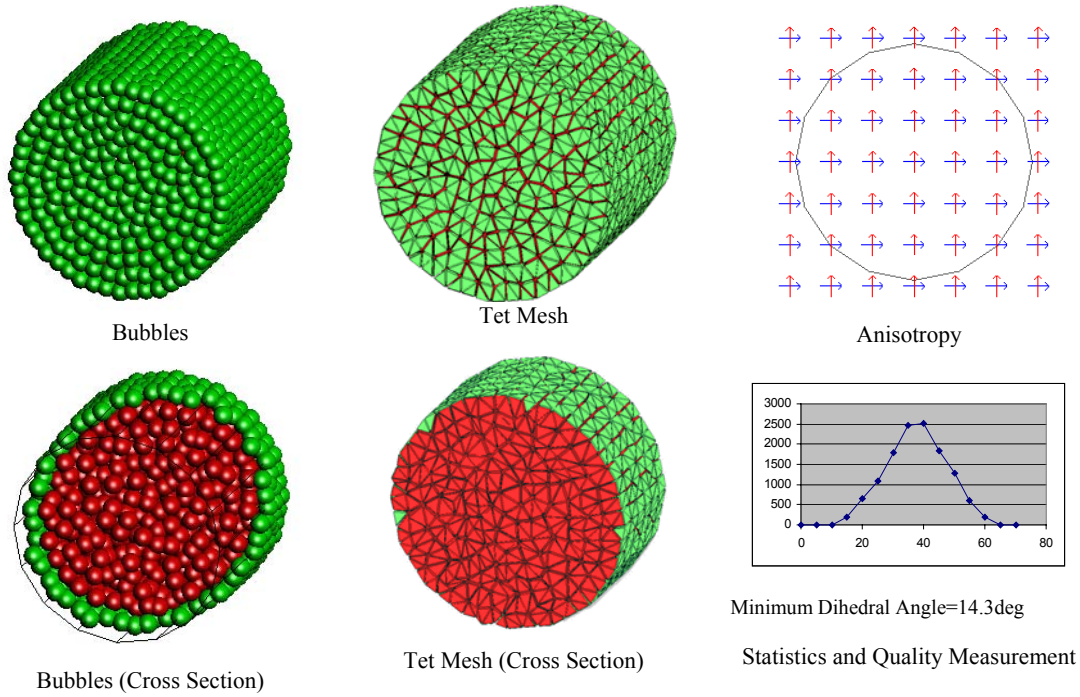
We also presented three experimental results to demonstrate the capability of our method. All three cases indicate that our method creates high quality meshes with elements that conform well to a given anisotropy.

## ACKNOWLEDGEMENTS

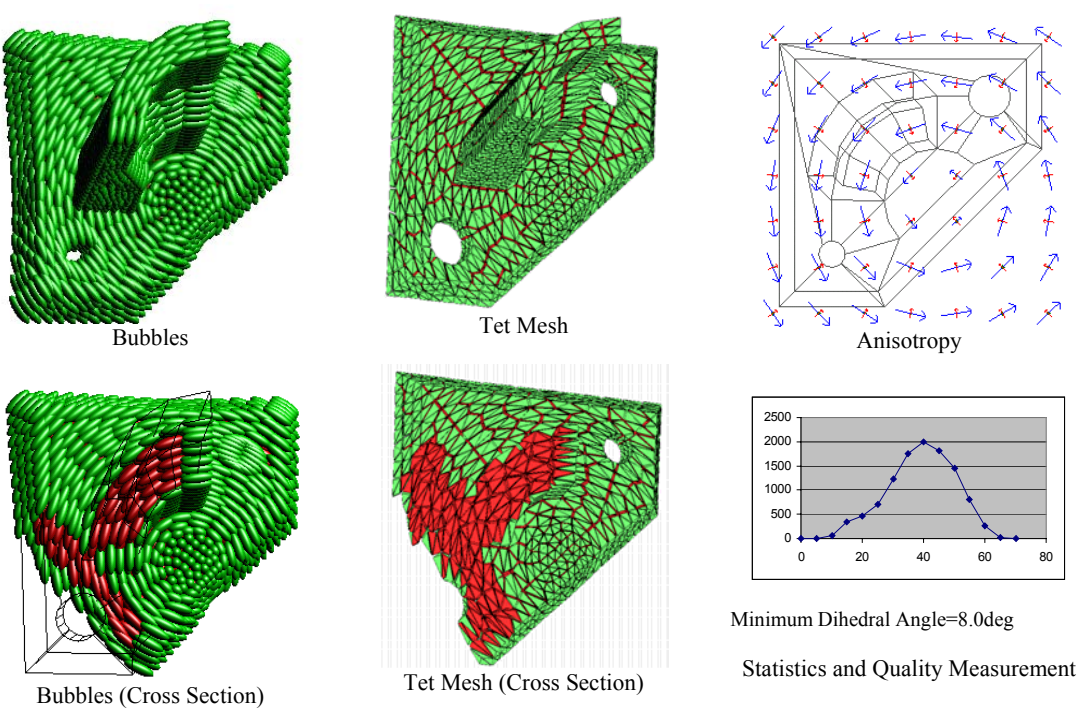
This material is based in part on work supported under a NSF CAREER Award (No. 9985288) and a grant from IBM Research.

We are grateful for the insightful technical discussions with A.Yamada, T.Itoh and K.Inoue of IBM Research in the development of this work.

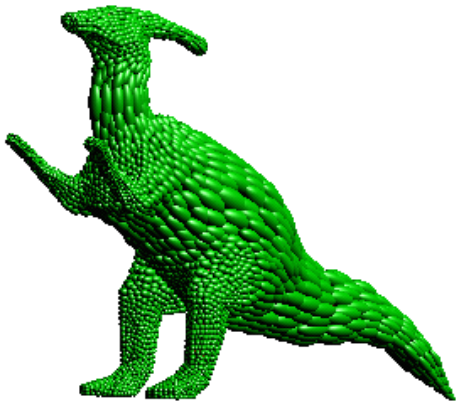




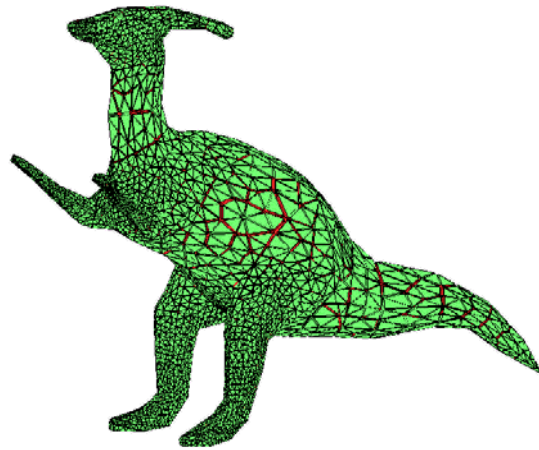
**Figure 10 Cylinder (12634 tets, 2406 nodes)**



**Figure 11 Mechanical Part (10921 tets, 2580 nodes)**



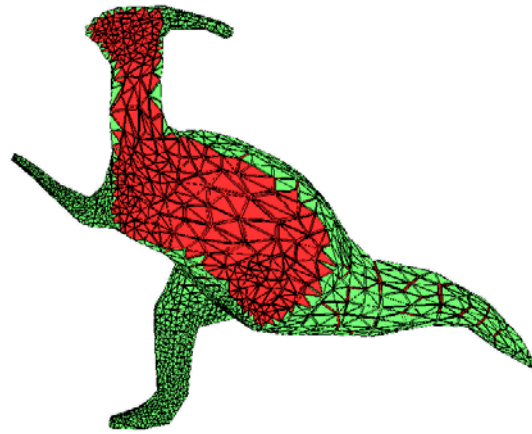
Bubbles



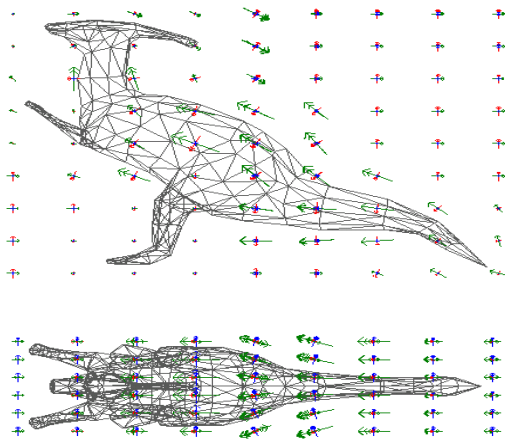
Tet Mesh



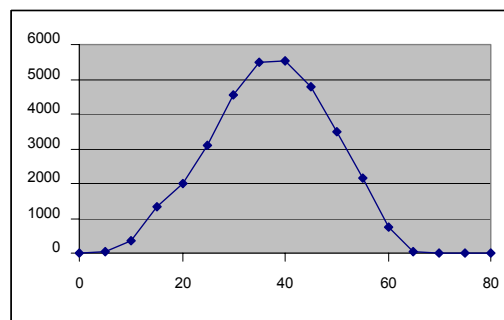
Bubbles (Cross Section)



Tet Mesh (Cross Section)



Anisotropy



Minimum Dihedral Angle=5.6deg

Statistics and Quality Measurement

Figure 12 Dinosaur (31069 tets, 7394 nodes)

## REFERENCES

- [1] Houman Borouchaki, Pascal J. Frey, Paul Louis George, "Unstructured Triangular-Quadrilateral Mesh Generation. Application to Surface Meshing," *Proceedings of IMR96*, pp.229-242, 1996
- [2] Frank J.Bossen, Paul S. Heckbert, "A Pliant Meshod for Anisotropic Mesh Generation," *Proceedings of IMR96*, pp.63-74, 1996
- [3] M.J.Castro-Diaz, F.Hecht, B.Mohammadi, "New Progress in Anisotropioc Grid Adaptation for Inviscid and Viscous Flows Simulations," *Proceedings of IMR95*, pp.73-85
- [4] Peter Fleischmann, Siegfried Selberherr, "Three-Dimensional Delaunay Mesh Generation Using a Modified Advancing Front Approach," *Proceedings of IMR97*, pp.267-278, 1997
- [5] Rao V. Garimella, Mark S. Shephard, Boundary Layer Meshing for Viscous Flows in Complex Domain, *Proceedings of IMR98*, pp.107-118,1998
- [6] Paul Louis George, "TET MESHING: Construction, Optimization and Adaptation," *Proceedings of IMR99*, pp.133-141, 1999
- [7] Barry Joe, "Construction of three-dimensional Delaunay triangulations using local transformations," *Computer Aided Geometric Design 8*, pp.123-142, 1991
- [8] Barry Joe, "Construction of three-dimensional improved-quality triangulations using local transformations," *SIAM Journal on Scientific Computing*, Vol. 16, No. 6, pp. 1292-1307, 1995
- [9] Xiang-Yang Li, Shang-Hua Teng, Alper Ungör, "Biting Ellipses to Generate Anisotropic Mesh," *Proceedings of IMR99*, pp. 97-108
- [10] Rainald Löhner, Juan R. Cebal, "Parallel Advancing Front Grid Generation," *Proceedings of IMR99*, pp.67-74
- [11] Kenji Shimada, "Physically-Based Mesh Generation : Automated Triangulation of Surfaces and Volumes via Bubble Packing," *Ph.D thesis, Massachusetts Institute of Technology*, Cambridge, MA, 1993
- [12] Kenji Shimada, David C. Gossard, "Bubble Mesh : Automated Triangular Meshing of Non-manifold Geometry by Sphere Packing," *In 3rd Shymposium on Solid Modeling and Applications*, pp. 409-419, 1995
- [13] Kenji Shimada, Atsushi Yamada, Takayuki Itoh, Anisotropic Triangular Meshing of Parametric Surfaces via Close Packing of Ellipsoidal Bubbles, *Proceedings of IMR97*, pp.375-390, 1997
- [14] Kenji Shimada, David C. Gossard, "Automatic Triangular Mesh Generation of Trimmed Parametric Surfaces for Finite Element Analysis," *Computer Aided Geometric Design*, Vol. 15, No. 3, pp. 199-222, 1998
- [15] D.F.Watson, "Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes," *The Computer Journal*, Vol.24, No.2, pp.167-172, 1981