RELIABLE ISOTROPIC TETRAHEDRAL MESH GENERATION BASED ON AN ADVANCING FRONT METHOD

Yasushi Ito[1], Alan M. Shih[2] and Bharat K. Soni[3]

University of Alabama at Birmingham, Birmingham, AL, U.S.A.
[1]yito@uab.edu, [2]ashih@uab.edu, [3]bsoni@uab.edu

ABSTRACT

In this paper, we propose a robust isotropic tetrahedral mesh generation method. An advancing front method is employed to control local mesh density and to easily preserve the original connectivity of boundary surfaces. Tetrahedra are created by each layer. Instead of preparing a background mesh for mesh spacing control, this information is estimated at the beginning of each layer at each node from the area of connecting triangles on the front and a user-specified stretching factor. An alternating digital tree (ADT) is prepared to correct the mesh spacing information and to perform geometric search efficiently. At the end of the mesh generation process, angle-based smoothing and Delaunay refinement are employed to enhance the resulting mesh quality. Surface meshes are prepared beforehand using a direct advancing front method for discrete surfaces extracted from computed tomography (CT) or magnetic resonance imaging (MRI) data. The algorithm is demonstrated with several biomedical models.

Keywords: mesh generation, advancing front method, unstructured, isotropic tetrahedral mesh, biomedical, computed tomography, CT, magnetic resonance imaging, MRI

1. INTRODUCTION

Thanks to advances in numerical algorithms and computer hardware, unstructured mesh approaches have been used for a wide range of computational filed simulations (CFS) and the simulations can be performed with reasonable computational accuracy and cost. For the volume mesh generation, isotropic tetrahedral meshes are widely used for inviscid and low Reynolds number viscous flows, structural/fracture analyses etc. due to their simplicity in terms of mesh generation. Therefore, the ability to generate a tetrahedral mesh from a given surface mesh automatically is essential in CFS.

Many papers have been published on the isotropic tetrahedral mesh generation. There are two major approaches: Delaunay triangulation methods [1 , 2] and advancing front methods [3-7]. The Delaunay triangulation is based on the empty sphere property. This mathematical criterion is a major backbone of the Delaunay approach. It enables high quality meshes and relatively fast runtime. The empty sphere property is, however, very sensitive to truncation errors in practical computations in 3D. More than five points are sometimes located on almost the same sphere. To resolve this problem, Shewchuk proposed an efficient algorithm for arbitrary precision floating-point arithmetic [8]. Integer coding for node coordinates is another approach. Although the conversion from real numbers to large integers slightly changes the node coordinates on the surface boundary, they are usually recovered at the end of the mesh generation process without any problem. However, a special treatment is still required for the nodes on the same spheres.

Boundary recovery is another issue needed to be resolved in the Delaunay approach. The empty sphere property does not ensure that the surface boundary preserves the original connectivity. The constrained Delaunay approach [9] cannot be easily extended to 3D complex domains. There are sometimes needs to generate a set of volume meshes for multi-connected domains, for example, to represent a different material property in each domain, or to generate hybrid meshes for viscous flow simulations. For parallel mesh generation, it is common to decompose the entire domain into a number of sub-domains and to tetrahedralize each sub-domain simultaneously. In this case, the shared internal boundaries must be identical. Although they can be changed during the parallel mesh generation process, the parallel efficiency often deteriorates due to the required communication between the processors.

The advancing front approach is based on the generation of tetrahedra by marching a front toward the interior. An initial front is usually defined as a water-tight surface mesh. Elements are created on the front, which is achieved by adding new points in the interior of the domain. This enables the generation of elements in variable size with desired stretching. Local mesh density near the initial front (i.e. boundaries) can be controlled easily. In addition, the topology of the initial front can be naturally preserved. The disadvantages of this approach are slow computational speed due to geometric search during the process and the quality of resulting meshes. Although these disadvantages must be resolved, there is no need to recover surface boundaries and the controllability is promising.

Combination of the advantages of the Delaunay and the advancing front approaches are widely used [10-12]. This algorithm starts with a Delaunay triangulation of a set of boundary nodes. This is used as a background mesh. New nodes are then added by the advancing front approach. This combined approach can shorten the runtime and produce high quality meshes. Local node density can be controlled easily. However, it still inherits the drawback of the Delaunay approach. Although the initial Delaunay triangulation is not difficult in 2D, the surface recovery is often troublesome in 3D.

Our approach is based on a simple advancing front method, and a Delaunay refinement method is employed at the end. The concept of the advancing front approach is clear from the analogy in 2D as well as the already published books [13] and papers. The problem is that they do not mention clearly about the detailed process in the tetrahedral mesh generation. One of the purposes of this paper is to discuss our approach to generate tetrahedral meshes using an advancing front method in detail. Nowadays, many commercial mesh generators support isotropic tetrahedral mesh generation. However, their reliability still needs to be discussed. Some of them force users to recreate the surface mesh again without providing adequate reason. Some do not support mesh generation for multi-connected domains.

In this paper, we focus on developing a robust tetrahedral mesh generation method based on an advancing front method. At the end of the mesh generation, an angle-based smoothing method and a Delaunay refinement method are applied. For the surface mesh generation, we use a direct advancing front method based on discrete surfaces. The method is applied to several biological geometries extracted from computed tomography (CT) or magnetic resonance imaging (MRI) data to demonstrate the capability of the system.

2. SURFACE MESH GENERATION

A GUI-based surface mesh generator has been developed based on the direct advancing triangulation method [14-16]. In this approach, a quality surface mesh is generated on an under-lying discrete surface called a background mesh. The background mesh should be a water-tight triangulated surface. If not, automated and manual repairing routines can be applied. Geometrical features are extracted from the background mesh. By specifying node distribution on the features, the local mesh density can be easily controlled. The advancing front method is then directly applied in 3D so as to ensure the quality of each triangle. For the geometric search, an alternating digital tree (ADT) is prepared in order to efficiently find neighboring nodes [17]. After the triangulation, an angle-based smoothing method is applied to the resulting mesh to ensure quality.

In this manner, the surface of interest is re-triangulated into a high-quality mesh to the desired degree. To use a mesh for numerical simulations, it should have isotropic elements and their sizes should be changed smoothly. Otherwise, the numerical error becomes significant and the simulations may fail.

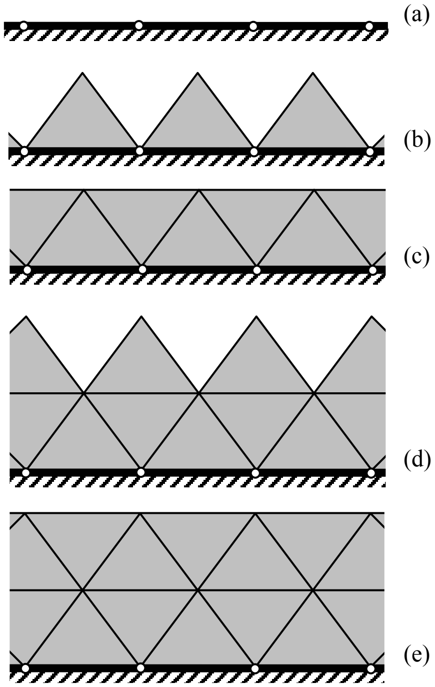


Figure 1. Tetrahedral mesh generation process: (a) part of initial front, defined as the 1st level front; (b) generating tetrahedra using triangles on the 1st level front; (c) generating tetrahedra using nodes on the 1st level front; (d) and (e) applying the same strategy to the 2nd level front

3. TETRAHEDRAL MESH GENRERATION

Figure 1 shows the outline of the tetrahedral mesh generation process. The same idea is discussed in Ref. 10. From the initial front defined by a surface mesh, tetrahedral elements are created inside the domain. To avoid complications in the front, tetrahedral elements are generated by each layer, which has a base front. Let us call it n-th level front ($n = 1, 2, \ldots$). When the n-th level front is completely constructed as shown in Figure 1a (initial front; 1st), Figure 1c (2nd) or Figure 1e (3rd), mesh spacing information is updated. Nodes to be added in accordance with the generation of tetrahedra belong to the (n + 1)-th level front, and the mesh spacing information is not prepared at the nodes until the (n + 1)-th level front is completely formed. The tetrahedral mesh generation is finished if the front has no triangles. Note that the front is not always simple like a water-tight surface mesh, whose edges have only two neighboring faces. Every edge on the front is expected to have 2^q ($q = 1, 2, \ldots$) neighboring faces.

During the mesh generation process, the quality of each tetrahedral element i is evaluated using the skewness ε_i.

$$\varepsilon_i = \frac{V_{i\mathrm{opt}} - V_i}{V_{i\mathrm{opt}}} \tag{1}$$

where V_i is the volume of tetrahedron i and $V_{i\mathrm{opt}}$ is the volume of an equilateral tetrahedron with the same circumradius. If ε_i is zero, the tetrahedron is equilateral. If it is one, it is degenerated.

The quality of a triangle can be evaluated in the same way.

$$\eta_i = \frac{a_{i\mathrm{opt}} - a_i}{a_{i\mathrm{opt}}} \tag{2}$$

where a_i is the area of triangle i and $a_{i\mathrm{opt}}$ is the area of an equilateral triangle with the same circumcircle.

3.1. Generating Tetrahedra

In this approach, we do not prepare a background mesh for the tetrahedral mesh generation to simplify the process. An influence region will be considered at each node on the n-th level front. The user input is a water-tight triangular surface mesh and a stretching factor for tetrahedral elements. The mesh generation process is summarized in the following subsections.

3.1.1. Required Data

Before discussing the method, the following data are maintained during the process.

- **Node *i***: coordinate $\mathbf{x}_i$
- **Node *i* on the *n*-th level front**: surface normal $\overline{\mathbf{N}}_i$ (Eq. 3); estimated height h'_{oi} (Eq. 6); searching radius r'_{si} (Eq. 7); number of remeshing
- **Node on the front**: connected triangles on the front
- **Tetrahedron**: 4 nodes
- **Triangle on the front**: 3 nodes; unit normal vector
- **ADT**: nodes on the front

3.1.2. Initial Front Setup

The mesh generation starts with inputting a water-tight surface mesh. The orientation of the faces must be the same (if not, see Ref. 16). The faces must face inside the domain; i.e., three nodes are stored in counter-clockwise order when each face is looked down upon from the inside as shown in Figure 2. The surface mesh is directly used as an initial front for the tetrahedral mesh generation. The initial front is also defined as the 1st level front.

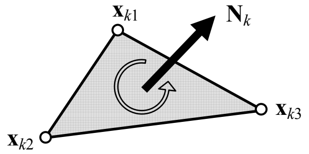


Figure 2. Face orientation

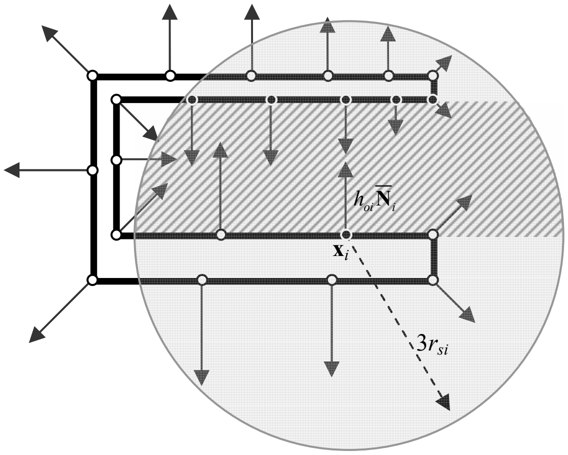


Figure 3. Mesh spacing information based on surface normals and area

3.1.3. n-th Level Front Setup

At each node $\mathbf{x}_i$ on the front, the following one vector and two values are prepared. They are illustrated in Figure 3. First, estimate a unit surface normal vector simply averaging the normals of the connected faces.

$$\overline{\mathbf{N}}_i = \sum_{k=1}^{m_k} \overline{\mathbf{N}}_k \Bigg/ \left| \sum_{k=1}^{m_k} \overline{\mathbf{N}}_k \right| \tag{3}$$

where m_k and $\overline{\mathbf{N}}_k$ denote the number of connecting triangles on the front and the unit normal of triangle k, respectively.

Second, define a searching radius r_{si} as the maximum length of the connected edges.

Third, calculate the reference height h_{oi}.

$$\begin{aligned} h_{oi} &= 1.24\alpha\sqrt{\overline{a}_i} \\ \overline{a}_i &= \left(\sum_{k=1}^{m_k} a_k \right) \Bigg/ m_k \end{aligned} \tag{4}$$

where a_k denotes the area of a neighboring triangle k. α is a user-specified stretching factor (the default value is 1.15). Suppose that the averaged area $\overline{a}_i$ is for an equilateral triangle. The constant 1.24 comes from the analogy for an equilateral tetrahedron ($2\sqrt{2}/\left(\sqrt{3}\sqrt[4]{3}\right)$).

h_{oi} is corrected considering an influence region at each node on the front based on the value r_{si} as shown in Figure 3. The influence region is defined as a sphere ($\mathbf{x}_j$, $3r_{sj}$) at node j. If node i is in the j's sphere, if $r_{sj} < r_{si}$, if $\cos^{-1}\left(\overline{\mathbf{N}}_i \cdot \overline{\mathbf{N}}_j\right) > 60^\circ$, and if node i is visible from node j, the coefficients c_i and c_j are updated as follows:

$$\begin{aligned} c_{oji} &= 0.05 \frac{3r_{si}}{\left|\mathbf{x}_j - \mathbf{x}_i\right|} \frac{h_{oj} - h_{oi}}{h_{oj}} \\ c_{ji} &= \min\left(\max\left(c_{oji},\ -0.6\right),\ 0.5\right) \\ c_i &= c_{ji} \ \text{ if } \left|c_{ji}\right| > \left|c_i\right| \\ c_j &= c_{ij} \ \text{ if } \left|c_{ij}\right| > \left|c_j\right| \end{aligned} \qquad (5)$$

The initial value for c_i is zero. In Figure 3, the visible region is the slashed zone and the black points represent the affected nodes. To find nodes in the specified sphere, geometric search is required. An efficient geometric search algorithm is essential for an advancing front method to shorten the runtime. For this purpose, an ADT is prepared, which registers all the nodes on the front.

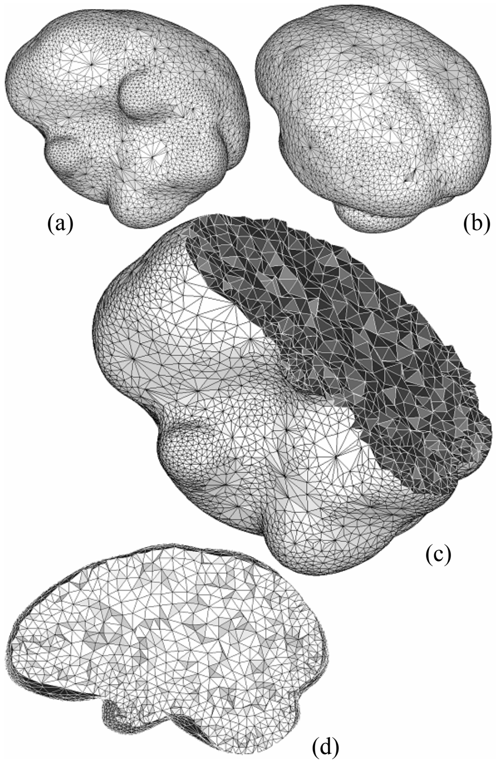


Figure 4. Tetrahedral mesh generation from a low-quality surface mesh: (a) and (b) surface mesh; (c) and (d) volume mesh showing a cross-section

Finally, the corrected reference height at node i is obtained after the influence from all the nodes on the front is considered.

$$h'_{oi} = \left(1.0 + c_i\right)h_{oi} \qquad (6)$$

This value should be smoothed using a Laplacian method considering the values at the neighboring nodes on the front in order to create tetrahedra locally having almost the same height. Note that the smoothing is also important not to create too small or too big elements and to prevent the mesh generation from failing. For example, Figure 4 shows a tetrahedral mesh from a low-quality surface mesh, which has slivers and large-sized jumps, just for demonstration purposes. Owing to the height smoothing, triangles appearing on the cross-section are locally uniform. The searching radius is also corrected as follows:

$$r'_{si} = \max\left(1.0 + c_i,\ 1.0\right)r_{si} \qquad (7)$$

3.1.4. Tetrahedral Mesh Generation Using Triangles on n-th Level Front

The beginning of this stage is illustrated in Figure 1a (1st) and Figure 1c (2nd) and the end is in Figure 1b and Figure 1d. The reason to select a base from the triangles on the n-th front is so that the height of each tetrahedron to be created can be controlled easily.

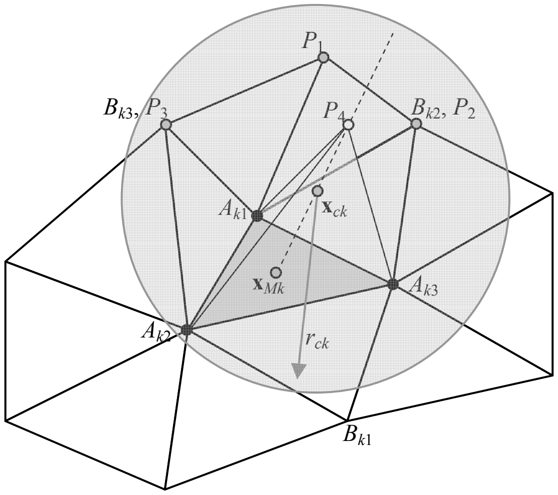


Figure 5. Selected base and candidates

1. Select triangle k that has the smallest area as a base for the tetrahedron to be created. Its apex should be selected from the already existing nodes or a newly created node as shown in Figure 5. All of them are stored in a candidate list, and the best node will be selected. Let A_{ki} (i = 1, 2, 3) be the three nodes of face k. Let a_k be its area. Face k has three neighboring triangles that share edges. Let B_{ki} (i = 1, 2, 3) be the node of the neighbor that is opposite to A_{ki}.

   All three nodes of face k are located on the n-th level front. The following averaged reference height is prepared based on Eq. 6.

$$h_k = \max\left(\frac{1}{3}\sum_{i=1}^{3} h'_{oi},\ \frac{1}{2}\sqrt{a_k}\right) \tag{8}$$

Index *i* denotes its three nodes. The searching radius at face *k* is defined as follows:

$$r_{sk} = 1.1 \times \max\left(r'_{s1},\ r'_{s2},\ r'_{s3},\ h_k\right) \tag{9}$$

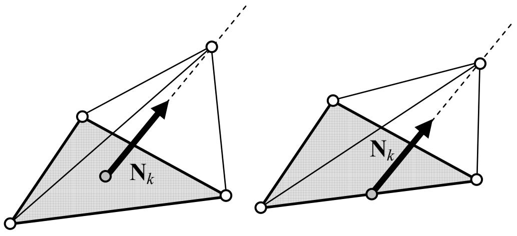
2. Determine new node position $\mathbf{x}_{pk}$. It can be calculated over the selected base as follows:

$$\mathbf{x}_{pk} = \mathbf{x}_{ok} + g h_k \overline{\mathbf{N}}_k \tag{10}$$

where $\mathbf{x}_{ok}$, $\overline{\mathbf{N}}_k$ and *g* denote a reference point position on face *k*, the unit normal vector of face *k*, and a coefficient (initially 1.0), respectively. h_k is from Eq. 8.

Some references suggest that the center of the face $\mathbf{x}_{Mk}$ be $\mathbf{x}_{ok}$ as shown in Figure 6a. If the triangle is almost equilateral, this is true. However, an advancing front method does not ensure that resulting tetrahedra and their faces are in good quality. Moreover, users may input a low-quality surface mesh. The reference point position should be changed considering the skewness of the triangle. If $\eta_k > 0.5$, the center of the longest edge is used as the reference point, instead of the center of the face, as show in Figure 6b.

In our approach, the candidate list has only one new node at first. If the new node is not a good candidate, try $g = 0.7^i$ (i = 1, 2, 3) in Eq. 10 later. Note that a polyhedron is not always tetrahedralized in 3D without adding a new node. Although an appropriate node can be added in this case, the new tetrahedron tends to be a sliver. It is better not to add such a new node. This issue is discussed later in Section 3.2.



(a) (b)

Figure 6. New node position x_{pk}: (a) based on the center; (b) based on the center of the longest edge

3. Select other possible candidates from the neighboring nodes on the front using the ADT. They must be in a certain sphere ($\mathbf{x}_{ck}$, r_{ck}) and be located over the face.

$$\begin{aligned} \mathbf{x}_{ck} &= 0.4\mathbf{x}_{Mk} + 0.6\mathbf{x}_{pk} \\ r_{ck} &= 1.1^{w} h_k \quad (w = 1, 2, \ldots, 10) \end{aligned} \tag{11}$$

where $\mathbf{x}_{Mk}$ is the center of the face. *w* will be increased if an appropriate node cannot be found after Step 4.

The candidates are sorted in ascending order by the value ζ_{kl}. The new point determined in Step 2 is listed at the end.

$$\zeta_{kl} = \left(\varepsilon_l + 0.1\right)\left|\mathbf{x}_{ck} - \mathbf{x}_l\right| \tag{12}$$

where index *l* and ε_l denote each candidate and the skewness of the tetrahedron defined by face *k* and candidate *l*, respectively. Now, we have *m* candidates and each of them is represented by P_l (l = 1, 2, … , m). P_m is a new point defined by Eq. 10.

4. Select a candidate P_l from the top of the list and perform intersection checks. For this purpose, neighboring nodes Q_ν in the sphere ($\mathbf{x}_{ck}$, r_{sk}) (see Eqs. 9 and 11) except A_{ki} (i = 1, 2, 3) and P_l should be prepared (ν = 1, 2, … , μ). Let T_l be the tetrahedron defined by face *k* and P_l.

- **Volume test**: the volume of T_l should be greater than $0.01 h_k a_k$.
- **Length test**: if the skewness of the face $\eta_k > 0.6$, the distance between $\mathbf{x}_{Mk}$ and P_l should be less than $\max\left(\left|\mathbf{x}_{ck}B_{k1}\right|,\ \left|\mathbf{x}_{ck}B_{k2}\right|,\ \left|\mathbf{x}_{ck}B_{k3}\right|,\ 2.0h_k\right)$. Otherwise, it should be less than $2.0h_k$.
- **Point in tetrahedron test**: nodes Q_ν should not be located in T_l.
- **Over the manifold test**: if edge $A_{ki}P_l$ (i = 1, 2, 3) has not been created, P_l must be located over the manifold at A_{ki}.
- **Edge-face intersection test (1)**: if edge $A_{ki}P_l$ (i = 1, 2, 3) has not been created, they should not intersect with existing faces on the front connected to nodes Q_ν. If face $A_{ki}A_{kj}P_l$ (i = 1, 2, 3 and j = 3, 1, 2) has not been created, it also should not intersect with existing edges on the front connected to nodes Q_ν. Note that the radius of the searching sphere r_{sk} is always greater than the local maximum edge length by definition. We just need to consider the neighbors of Q_ν for the edge-face intersection test.
- **Edge-face intersection test (2)**: If edge $A_{ki}P_l$ (i = 1, 2, 3) has not been created, it should not intersect with existing faces on the front connected to the two nodes A_{kj} (j = 1, 2, 3 and $j \neq i$).
- **Face-face (dihedral) angle test**: If face $A_{ki}A_{kj}P_l$ (i = 1, 2, 3 and j = 3, 1, 2) has not been created and if P_l and $B_{k(6-i-j)}$ are not the identical, check the dihedral angle between face $A_{ki}A_{kj}P_l$ and $A_{kj}A_{ki}B_{k(6-i-j)}$. The angle should be greater than 5°.
- **Edge-face angle test**: If edge $A_{ki}P_l$ (i = 1, 2, 3) has not been created, the angle between it and the manifold of the front at A_{ki} should be greater than 5°. If face $A_{ki}A_{kj}P_l$ (i = 1, 2, 3 and j = 3, 1, 2) has not been created, the angle between the face and the

edges on the front connected to A_{ki} and A_{kj} should be greater than 5°.

If all the tests are passed, proceed to the next step. If not, try the next candidate. If all the candidates in the list are examined, go back to Step 3 and increase w. if w = 10, perform remeshing around the selected triangle discussed later in Section 3.2.

5. Add a new tetrahedron, and a new node and faces if required, and update the front information and the ADT. If there is no triangle on the n-th front, go to the next section. If not, go back to Step 1.

3.1.5. Tetrahedral Mesh Generation Using Nodes on n-th Level Front

The beginning of this stage is illustrated in Figure 1b (1st) and Figure 1d (2nd) and the end is in Figure 1c and Figure 1e. The process is almost the same as that discussed in Section 3.1.4. The way to select the next base is different. All the three nodes of a face are not always on the n-th level front. In Step 1, Eqs. 8 and 9 should be estimated only using the nodes on the n-th level front. At least one node belongs to the n-th level front. If nodes on the n-th level front have no faces, proceed to the next level front. If the front has no triangles, the mesh generation process is completed.

3.2. Local Remeshing

During the mesh generation process, nodes sometimes cannot be added without creating extremely low-quality elements. In this case, it is better to remove the tetrahedra near the selected base and to remesh the resulting larger void in order to create higher quality elements. There are two reasons. First is that a node placed too close to a face is sensitive to truncation errors. In our approach, double precision floating-point numbers are used for real numbers without any special arithmetic algorithms to enable a cross-platform application easily, and hence this situation should be avoided. Second is that low-quality elements often come with low-quality faces, which produce low-quality elements again later. The mesh generation process may fail in the end.

A node on the n-th level front stores the number of remeshing process that affected it. If the remeshing is locally the first time, remove the tetrahedra on the n-th level front connected A_{ki} (i = 1, 2, 3). In case of more than two times, the removed region becomes large accordingly. If no tetrahedra on the n-th level front are found around the nodes, the remeshing should consider all the local tetrahedra. In connection with removing the tetrahedra, the front and the ADT should be updated.

3.3. Smoothing

Compared to the Delaunay approach, an advancing front method tends to generate low-quality elements. During the stage of the node insertion discussed in Section 3.1, more restrictions can be added not to create slivers. The problem is that they are sometimes computationally very expensive. Thus, we refine the resulting meshes at the end of the process combining three methods: a) node removal; b) angle-based node smoothing to optimize node locations; and c) Delaunay refinement to optimize connectivity of nodes. The three methods are applied in this order. The angle-based node smoothing and the Delaunay refinement are applied three times consecutively.

3.3.1. Node Removal

During the tetrahedral mesh generation process, some nodes may be assigned too small number of elements as shown in Figure 7. A node looks like in a tetrahedron if its neighbors are four (Figure 7a) or in a pyramid if its neighbors are six (Figure 7b). Since the neighbors of such nodes are often low-quality tetrahedra, the nodes are removed without any conditions.

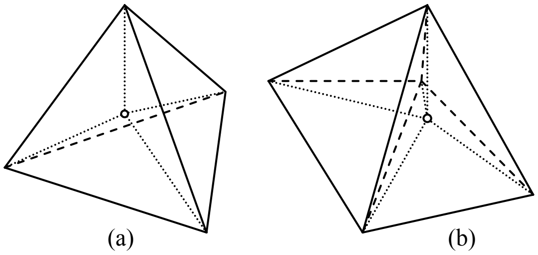


Figure 7. Needless nodes having (a) four neighbors; and (b) six neighbors

3.3.2. Angle-Based Node Smoothing

An angle-based smoothing method improves mesh quality by shifting node positions [18, 19]. Laplacian smoothing is often employed for enhancing mesh quality because it requires a low computational cost. However, it does not guarantee to improve mesh quality and often creates lower quality or invalid elements. The size and shape quality of the mesh after the angle-based smoothing method is much better than that after the Laplacian smoothing.

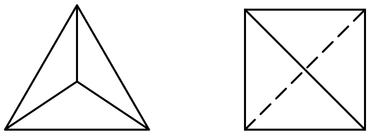


Figure 8. Overhead view of cap and sliver

3.3.3. Delaunay Refinement

Delaunay refinement has a strong mathematical background, and it improves the quality of meshes significantly. Considering our approach, most of the low-quality elements are caps and slivers as shown in Figure 8. Hence we applied a Delaunay refinement method as follows:

1. Select a tetrahedron whose skewness ε_i is greater than 0.3.
2. List its four nodes in descending order of sizes of the faces opposite to them, which correspond to the likelihood of dissatisfaction with the Delaunay empty sphere property.
3. Select a node A from the top. Consider a tetrahedron T_1 connected to the node and one of its faces F opposite to the node. If the circumsphere of the other tetrahedron T_2 that shares the face contains the node A, Delaunay refinement can be applied here. The following three swapping types are considered [20]:

   - Edge suppress
   - 2-3 (3-2) swap
   - 2-2 swap

   If a set of new tetrahedra after one of the swapping methods have better quality in terms of the skewness (i.e., the largest skewness of the new tetrahedralization is smaller than that of the previous tetrahedralization), the new tetrahedralization will be accepted. First, the edge suppress is utilized to create the edge AB, where B is the node of the tetrahedron T_2 opposite to the face F. If the edge cannot be created or the new tetrahedralization is not accepted, the face F is swapped.
4. Step 3 is continued until no face swapping is occurred around the node.

Note that this approach does not follow the empty sphere property strictly in order to retain the original connectivity on the surface mesh and to avoid truncation errors. The quality of tetrahedral meshes is, however, improved significantly. An example will be shown in Section 4.2.

4. APPLICATIONS

Background meshes for the surface triangulation are extracted from CT/MRI data to represent high-fidelity models. The tetrahedral mesh generation method has been applied to brain/skull models and a pelvic bone.

4.1. Simplified Brain/Skull Model

Computational simulations for traumatic brain injury (TBI) are required to understand how it occurs and, for example, to develop safer cars. A set of meshes for the simplified brain/skull model is shown in Figure 9. Major components of a head are skin, bone (skull and spine), brain (white and gray matters) and cerebrospinal fluid (CSF). The modeling of CSF is especially important because CSF acts to cushion a blow to the head and lessen the impact. In this case, meshes are generated for the brain (28,932 nodes and 149,268 tetrahedra), the region between the brain and skull (Brain-Inner; 29,838 nodes and 127,062 tetrahedra), and the skull (Inner-Outer; 20,301 nodes and 71,992 tetrahedra).

An advancing front method easily preserves the connectivity of boundary surfaces. Although three tetrahedral meshes are generated for the three different domains, the corresponding surface boundaries are the identical. Figure 10 shows skewness of the tetrahedra for the three meshes. Although the outermost two layers have thin volumes, the quality of the elements is acceptable.

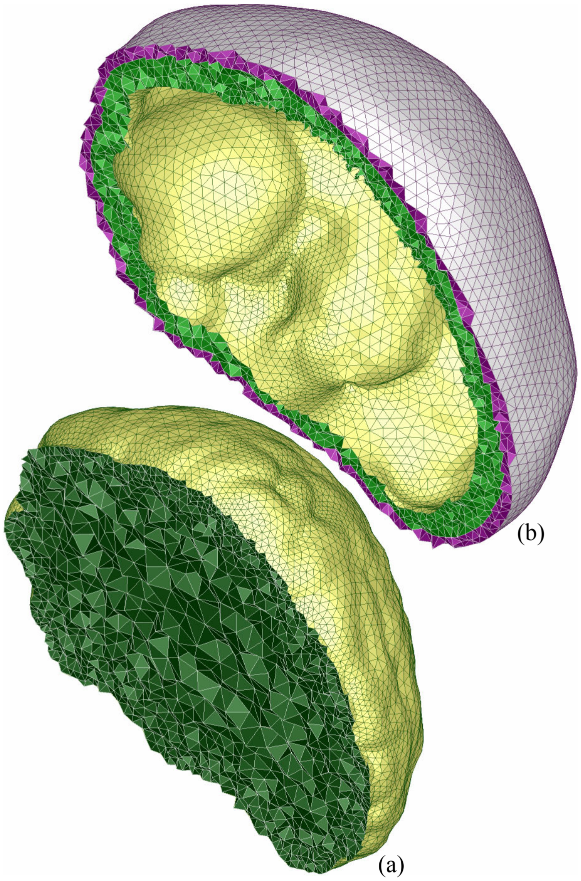


Figure 9. Set of tetrahedral meshes for brain/skull model (half): (a) brain model based on MRI data; (b) gap between skull and brain (the cross section indicated by green) and skull model (by purple)

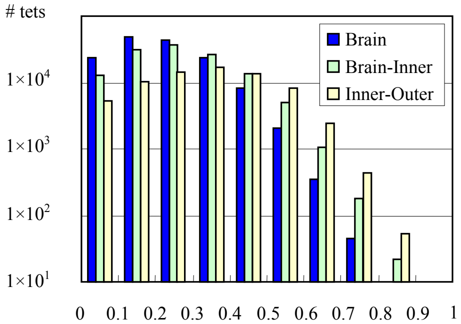


Figure 10. Skewness of the tetrahedra in the simplified brain/skull model

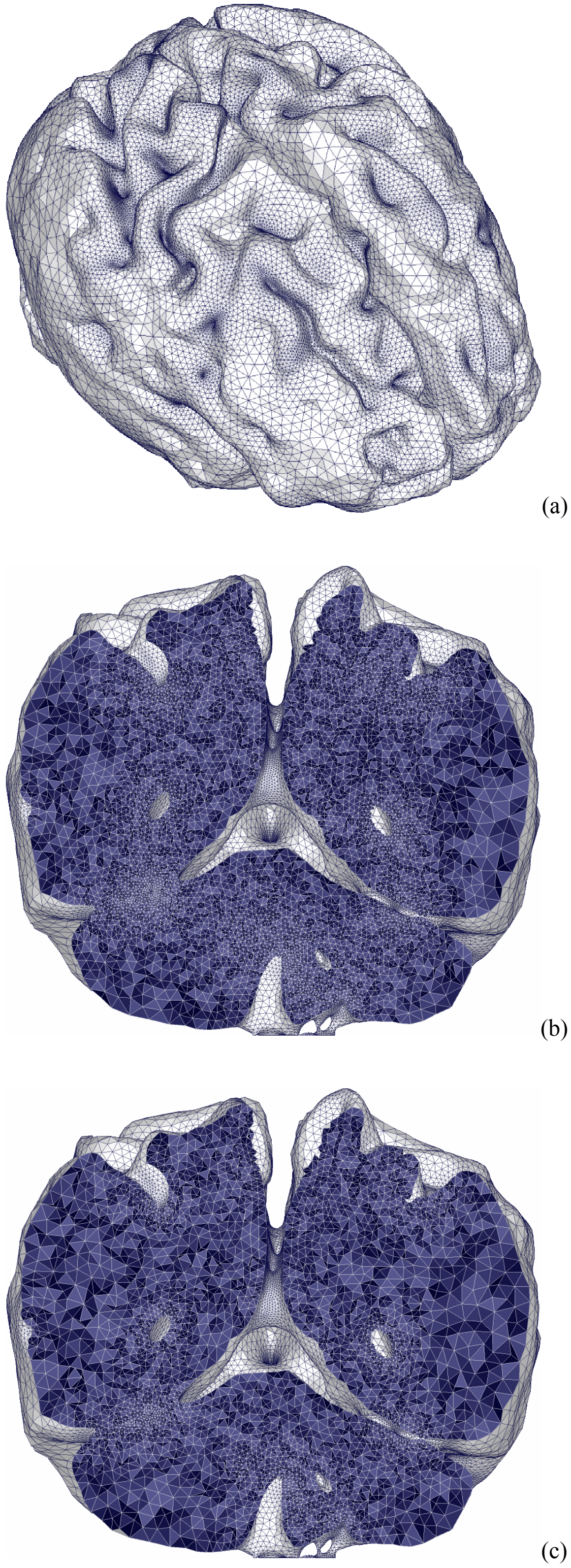


Figure 11. Tetrahedral mesh for detailed brain model: (a) surface mesh; (b) cross section (α = 1.05); (c) cross section (α = 1.15)

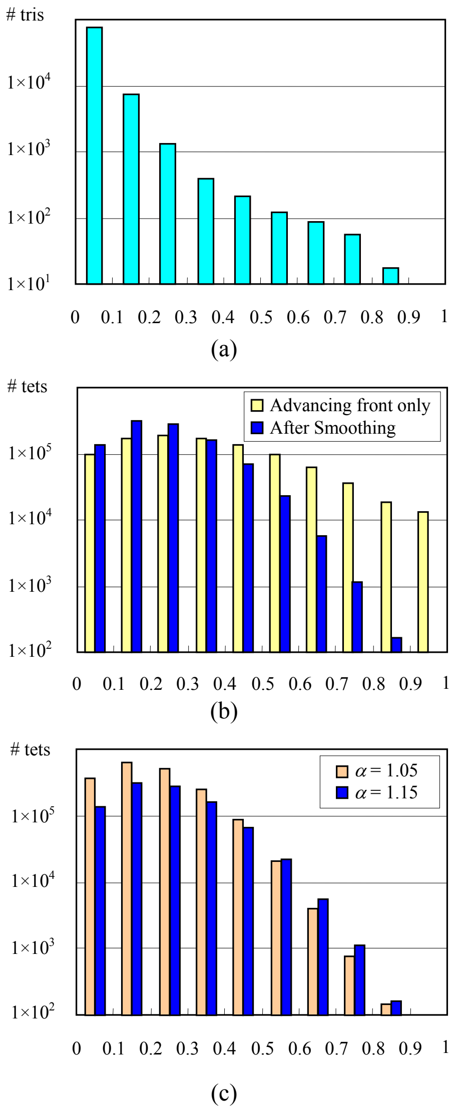
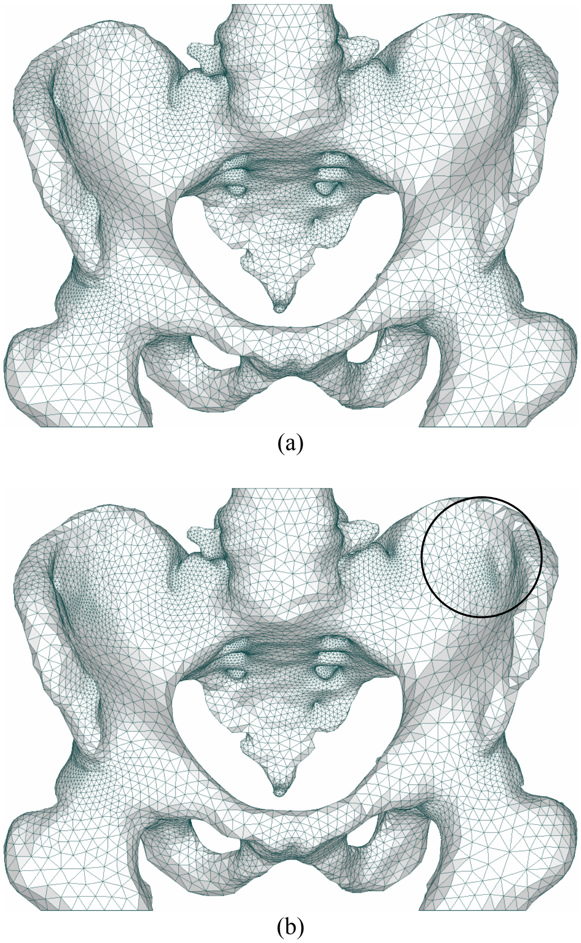


Figure 12. Skewness of the elements in the detailed brain model: (a) surface mesh; (b) tetrahedral meshes before and after the smoothing (α = 1.15); (c) tetrahedral meshes (α = 1.05 and 1.15)

4.2. Detailed Brain Model

Figure 11 shows two tetrahedral meshes for a detailed brain model based on MRI data. The surface mesh contains 84,510 boundary triangles. The volume meshes have 340,492 nodes and 1,896,578 tetrahedra when the stretching factor α = 1.05; 185,958 nodes and 982,004 tetrahedra when α = 1.15. Figure 12 shows skewness of the elements based on Eqs. 1 and 2 (triangles in the surface mesh and tetrahedra in the volume meshes). In this case, the number of elements needs to be as small as possible in spite of the detailed model. Consequently, the surface mesh has several low-quality elements. The smoothing methods discussed in Section 3.3 improve the quality of the tetrahedral mesh significantly as shown in Figure 12b. The elements whose skewness is more than 0.8 are 0.010% when α = 1.05 and 0.018% when α = 1.15. Even if the stretching factor becomes large, the number of elements is reduced greatly and the mesh quality is still excellent as shown in Figure 12c.

The tetrahedral mesh generation method is robust even if the input surface mesh is complicated and has some low-quality elements. Cross-sections at the same location are rendered in Figure 11b and Figure 11c. Smooth size variation can be seen. The quality of the mesh is enough to use numerical simulations.



(a)

(b)

Figure 13. Surface mesh generation for a human pelvis bone: (a) original; (b) after refinement considering thickness of the volume

4.3. Human Pelvis Bone

To investigate the injury mechanism of human bones, mesh generation for them from CT/MRI data is required to perform realistic simulations. Figure 13 shows surface meshes for a human pelvis bone. The quality of the original surface mesh shown in Figure 13a is excellent if a volume mesh is created outside. In actual numerical simulations, reducing the number of elements is sometimes important, while the quality of the volume mesh is also important. To create a better volume mesh, the surface mesh is refined considering local thickness of the volume beforehand as shown in Figure 13b. The surface mesh has 22,045 nodes and 44,186 faces. The volume mesh shown in Figure 14 has 58,115 nodes and 360,212 tetrahedra. Figure 15 shows skewness of the elements.

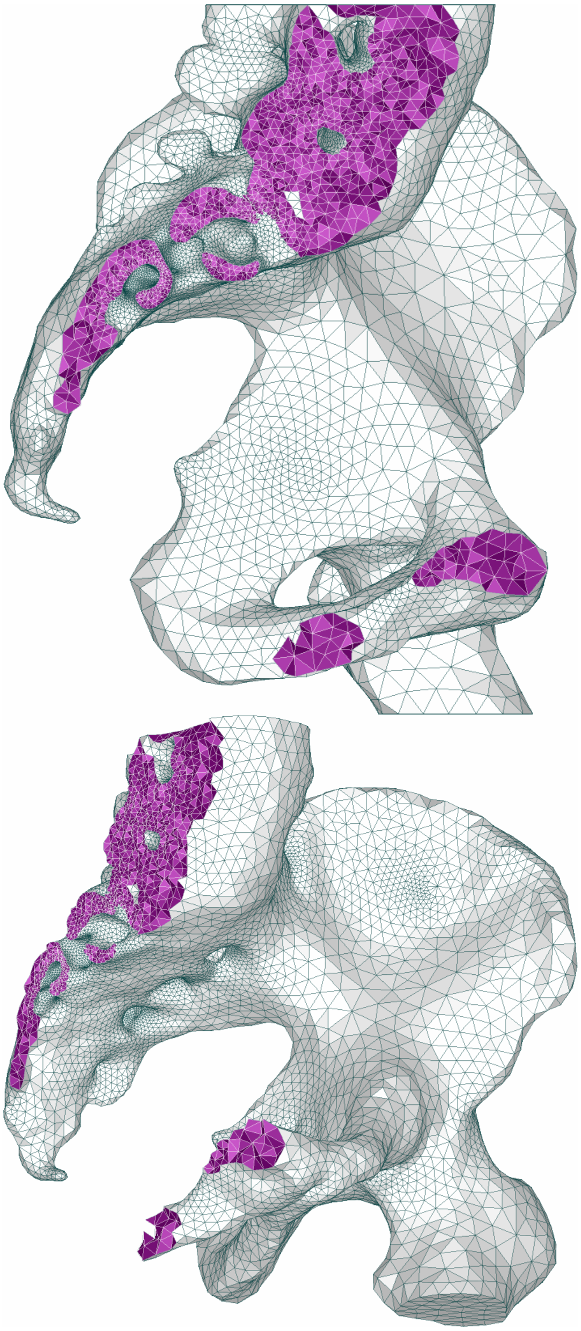
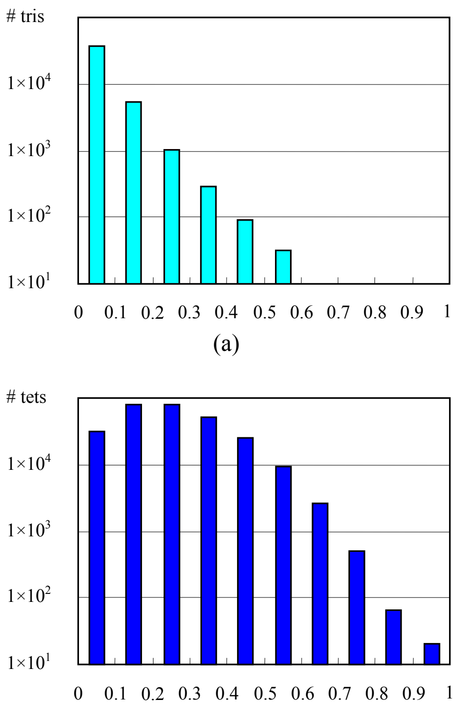


Figure 14. Tetrahedral mesh generation for a human pelvis bone (a cross-section indicated by purple)



(a)

(b)

Figure 15. Skewness of the elements (pelvis bone): (a) surface mesh; (b) tetrahedral mesh

5. CONCLUSIONS

A robust isotropic tetrahedral mesh generation method is presented. An advancing front method is employed for the tetrahedrization. Instead of preparing a background mesh for mesh spacing control, tetrahedra are created by each layer and mesh spacing information is estimated at the beginning of the layer from the area of triangles on the front and a user-specified stretching factor. An alternating digital tree (ADT) is prepared to correct the spacing information, and to perform geometric search efficiently. The remeshing routine is implemented during the process so as to not create extremely low-quality elements. At the end of the mesh generation process, an angle-based smoothing method and a Delaunay refinement method are employed to enhance the quality of resulting meshes. The advancing front method proposed here is robust even if the input surface mesh contains low-quality triangles. The tetrahedral mesh generation method is applied to several biological geometries based on CT/MRI data. The nature of the advancing front method enables no connectivity changes for the input surface meshes and the controllability for local mesh density. The smoothing methods based on the angle-based smoothing and Delaunay refinement ensure the quality of resulting meshes. The quality of resulting meshes is excellent and suitable for numerical simulations.

ACKNOWLEDGEMENTS

This research is supported in part by NASA URETI Program and NSF ITR Adaptive Software Project. The authors would like to thank Dr. Nikos Chrisochoides and Mr. Andrey Chernikov who provided valuable feedbacks and suggestions to improve the robustness of the tetrahedral mesh generator. The authors would also like to acknowledge the contribution from Mr. Corey Shum for creating discrete surfaces from CT/MRI datasets.

REFERENCES

[1] Baker, T. J., “Shape Reconstruction and Volume Meshing for Complex Solids,” *International Journal for Numerical Methods in Engineering*, Vol. 32, 1991, pp. 665-675.

[2] Sharov, D. and Nakahashi, K., “A Boundary Recovery Algorithm for Delaunay Tetrahedral Meshing,” Proceedings of the 5th International Conference on Numerical Grid Generation in Computational Field Simulations, Mississippi State University, MS, 1996, pp.229-238.

[3] Peraire J., Peiro J., Formaggia L., Morgan K. and Zienkiewicz O. C., “Finite Element Euler Calculations in Three Dimensions,” *International Journal for Numerical Methods in Engineering*, Vol. 26, 1988, pp. 2135-2159.

[4] Löhner, R. and Parikh, P., “Generation of Three-Dimensional Unstructured Grids by the Advancing Front Method,” *International Journal for Numerical Methods in Fluids*, Vol. 8, 1988, pp. 1135-1149.

[5] Morgan, K., Peraire, J. and Peiró, J., “Unstructured grid methods for compressible flows,” *AGARD Report 787, Special Course on Unstructured Grid Methods for Advection Dominated Flows*, 1992, 5.1-5.39.

[6] Jin, H. and Tanner, R. I., “Generation of Unstructured Tetrahedral Meshes by the Advancing Front Technique,” *International Journal for Numerical Methods in Engineering*, Vol. 36, 1993, pp. 1805-1823.

[7] George, P. L. and Seveno, E., “The Advancing-Front Mesh Generation Method Revisited,” *International Journal for Numerical Methods in Engineering*, Vol. 37, 1994, pp. 3605-3619.

[8] Shewchuk, J. R., “Adaptive Precision Floating-Point Arithmetic and Fast Robust Geometric Predicates,” *Discrete & Computational Geometry*, Vol. 18, 1997, pp. 305-363.

[9] Chew, L. P., “Constrained Delaunay Triangulations,” *Algorithmica*, Vol. 4, No. 1, 1989, pp. 97-108.

[10] Müller, J. D. Roe, P. L. and Deconinck, H., “A Frontal Approach for Node Generation in Delaunay Triangulations,” *AGARD Report 787, Special Course on Unstructured Grid Methods for Advection Dominated Flows*, 1992, 9.1-9.7.

[11] Mavriplis, D. J., “An Advancing Front Delaunay Triangulation Algorithm Designed for Robustness,” *Journal of Computational Physics*, Vol. 117, 1995, pp. 90-101.

[12] Marcum, D. L. and Weatherill, N. P., “Unstructured Grid Generation Using Iterative Point Insertion and Local Reconnection,” *AIAA Journal*, Vol. 33, No. 9, 1995, pp.1619-1625.

[13] Thompson, J. F., Soni, B. K., Weatherill, N. P., *Handbook of Grid Generation*, CRC Press, Boca Raton, 1999.

[14] Nakahashi, K. and Sharov, D., “Direct Surface Triangulation Using the Advancing Front Method,” AIAA Paper 95-1686-CP, 1995, pp.442-451.

[15] Ito, Y. and Nakahashi, K., “Direct Surface Triangulation Using Stereolithography Data,” *AIAA Journal*, Vol. 40, No. 3, 2002, pp. 490-496.

[16] Ito, Y. and Nakahashi, K., “Surface Triangulation for Polygonal Models Based on CAD Data,” *International Journal for Numerical Methods in Fluids*, Vol. 39, Issue 1, 2002, pp. 75-96.

[17] Bonet, J. and Peraire, J., “An Alternating Digital Tree (ADT) Algorithm for 3D Geometric Searching and Intersection Problems,” *International Journal for Numerical Methods in Engineering*, Vol. 31, 1991, pp. 1-17.

[18] Zhou, T. and Shimada, K., “An Angle-Based Approach to Two-dimensional Mesh Smoothing,” Proceedings of the 9th International Meshing Roundtable, 2000, pp.373-384.

[19] Ito, Y. and Nakahashi, K., “Improvements in the Reliability and Quality of Unstructured Hybrid Mesh Generation,” *International Journal for Numerical Methods in Fluids*, Vol. 45, Issue 1, 2004, pp. 79-108.

[20] Owen, S. J. and Saigal, S., “H-Morph: an Indirect Approach to Advancing Front Hex Meshing,” *International Journal for Numerical Methods in Engineering*, Vol. 49, Issue 1-2, 2000, pp. 289-312.