Constrained Tetrahedral Mesh Generation of Human Organs on Segmented Volume *

Xiaosong Yang[1], Pheng Ann Heng[2], Zesheng Tang[3]
[1]Department of Computer Science and Technology, Tsinghua University, Beijing 100084, 86-10-62782953, P.R. China, yangxiaosong@yahoo.com
[2]Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, pheng@cse.cuhk.edu.hk
[3]Macau University of Science and Technology, ztang@must.edu.mo

Abstract

Virtual endoscopy is a new method of diagnosis using computer processing of 3D image datasets (such as CT or MRI scans) to provide simulated visualization. In order to obtain a physically realistic surgery simulation, it is needed to generate the accurate 3D meshes for finite element analysis (FEA) to simulate serials of actions in the surgery. In this paper, a new algorithm is proposed to create the tetrahedral mesh directly from the segmented volume. The adaptive model generated has the attributes of accurate, small scale and well-shaped which is very suitable for complete 3D finite element solvers.

Keywords: Tetrahedralization, Finite Element Analysis

1. Introduction

As the result of segmentation on serials of CT or MRI images, we get a volume of flags which tag the tissue type of each voxel. Fig 1 gives a 2D slice of a simple tagged volume which has only two tissue types.

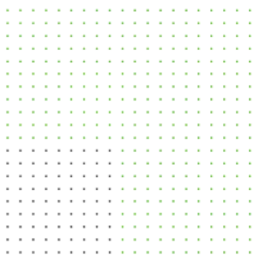


Fig 1. 2D slice of tagged volume

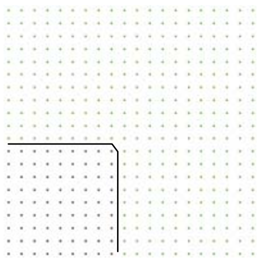


Fig 2. Isosurfaces on the sample volume

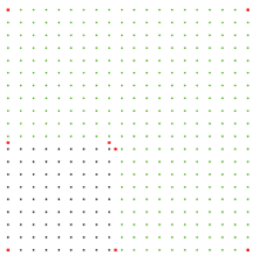


Fig 3. Featured point

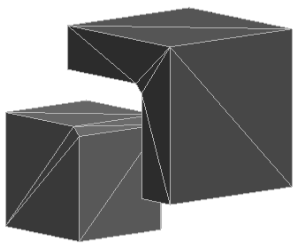


Fig 4. Delaunay tetrahedralization

*Sponsored by the Joint Project of The National Natural Science Foundation of China (Grant number: 6001161942) and RGC of Hong Kong (Grant number is N_CUHK412/00).

There are two general methods to create the tetrahedral mesh on the segmented volume. Interval volume tetrahedralization[1] is a good example of the first category. It decomposes the generated interval volume in each voxel into tetrahedron. Because the tetrahedron size is smaller than the voxel, the generated 3D mesh is always too big to meet the requirement of real time FEA. It needs a post-processing step to simplify the mesh. Another method[2] generates isosurfaces to meet the requirement of 3D conforming Delaunay tetrahedralization algorithm. The isosurfaces are taken as the constraining faces. Fig 2 illustrates the 2D slice of isosurfaces generated on the sample volume. The advantage of this method is that it keeps the detailed boundary of different tissues. But it also suffers from too fractal triangles on the boundary. And when two adjacent tissues adhere to each other tightly, the algorithm will cause holes or overlap at the interface as the result of mesh decimation. Because the simple cases considered by Marching Cube, it can not create the correct isosurfaces when a voxel contains more than two tissue types.

Our method is different from the above two methods. But it keeps the tissues' accurate geometrical features just like the second algorithm. The generated tetrahedral mesh has the following attributes:

(1) Accurate. The tissue boundary contained in the original dataset is kept accurately by the featured point selection. We consider it as a constrained problem.

(2) Small scale. Because FEA is a very computational intensive process. It is impossible to simulate a large-scale dataset in real time. An automatic self-adaptive method is presented to vary the density of mesh nodes according to local features of the segmented volume.

(3) Well-shaped. The quality of 3D mesh takes a very important role for the accuracy of FEA computation. A well-shaped mesh presents less error in the simulation.

2. Tetrahedral Mesh Generation

Because Delaunay triangulation guarantees the well-shape of the final mesh. We follow the idea and classify our method as an incremental insertion algorithm in Delaunay triangulation category. It is composed of three phases:

1. Placement of the mesh vertices. This phase would definitely affect several qualities of the final meshes, such as the adaptiveness of mesh density, the conformation to the tissue boundaries etc. In this paper, we take three steps to meet these targets: coarse structured mesh, feature point selection and Steiner point displacement.
2. The second phase is Delaunay Triangulation. The Delaunay tetrahedralization of a point set is defined by the empty sphere condition. We will combine the incremental insertion and flipping based algorithm to generate the final mesh.
3. Restore the tissue boundary and set element's tissue type. After the triangulation, the boundary of different tissues may be not guaranteed according to the empty sphere condition. In this paper, three basic flip operations are used to restore the boundary. Each element's tissue type is not so easy to be determined especially when the element is on the tissue boundary. It will depend on which side the element is on according to the 'isosurfaces'.

3. Point Displacement

There are three kinds of points for incremental insertion: feature point, steiner point and structured mesh point.

Feature points should include all points that represent the tissue’s geometrical shape accurately. They take the responsibility of conformation to the embedded tissue boundary. The displacement of steiner point should take care about the adaptive mesh density. The structured mesh points are selected to solve the insertion failure problem especially when the volume size is too big.

3.1 Feature point

The most important concept of our method is the feature point. The feature point is the voxel edge midpoint which has abrupt gradient in the local neighbors.

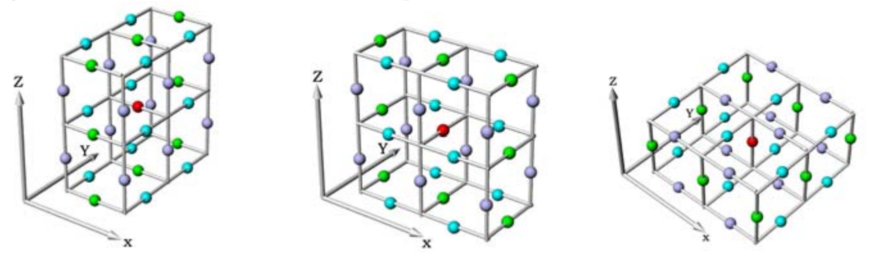
Fig 3 shows the feature points on the slice of the sample volume. Normally these feature points are the exact vertex of the simplified isosurfaces triangles. Using these feature points, we can get a very simple tetrahedral mesh by 3D Delaunay tetrahedralization algorithm. But the quality of the mesh is not very good, especially at the neighbors of the feature points that are very close to each other.

The computation of feature points are pretty complicate. It takes three steps:

1. Gradient computation of the mid point of each voxel edge
2. Compare of the gradient in the local neighbors
3. Error bounded simplification of feature point

There are three kinds of mid points, x+0.5, y+0.5, z+0.5 which lie on the voxel edges aligned to x, y and z axis respectively. We can simply use linear interpolation to get the gradient from its two neighbor voxel nodes. In order to include the isosurfaces point on the volume surfaces, we extend the volume one slice at the two ends in each axis direction to contain an impossible tissue type in the volume segmentation.

The neighborhood of a mid point in the gradient compare step should take all possible directions of isosurfaces (Fig 5). If the mid point does not have the same gradient value in its neighborhood, it is selected as a feature point.



x+0.5 neighbors y+0.5 neighbors z+0.5 neighbors

Fig 5 Neighborhood of a mid point to compare gradient

In the isosurfaces simplification method, we always take an error bound into account. Here we can also define a global error limit to simplify feature points in the local neighbors. If two featured points have following attributes, they can be merged together:

1. Accessible to each other
2. Boundary of the same tissue pairs.
3. Gradient difference less than the global error bound.

3.2 Steiner point displacement

We can get a coarse tetrahedral mesh using these feature points. Then using some smooth algorithms to improve mesh quality. But ordinarily the mesh quality is very bad at this stage (Fig 4), this should take a lot of iteration steps to achieve a satisfied mesh. And it is very hard to control the adaptiveness of the mesh density. Instead, we present a method to generate some inner Steiner points to give a better point displacement.

We define a global density field to control the automatic point displacement. This can be taken as a problem of interpolation of scattered point. We can easily define a density value D_k for each featured point. It is based on the minimal distance from this point to all other featured points in the local area. So for each voxel node in the volume, we can get the density value by using inverse distance weighted interpolation methods:

$$D(x,y,z) = \sum_{k=1}^{n} D_k \bullet \frac{\prod_{j \neq k} d_j(x,y,z)}{\sum_{k=1}^{n} \prod_{j \neq k}^{n} d_j(x,y,z)} \tag{1}$$

Where $d_k(x,y) = \sqrt{(x-x_k)^2 + (y-y_k)^2}$ is the distance between point (x,y) to (x_k,y_k). For each voxel node in the considered area, if there is no other points in the sphere centered on the point with radius of density, this point is inserted into the point queue to generate final meshes.

Figure 6 shows the points displacement on a 2D slice of the sample volume.

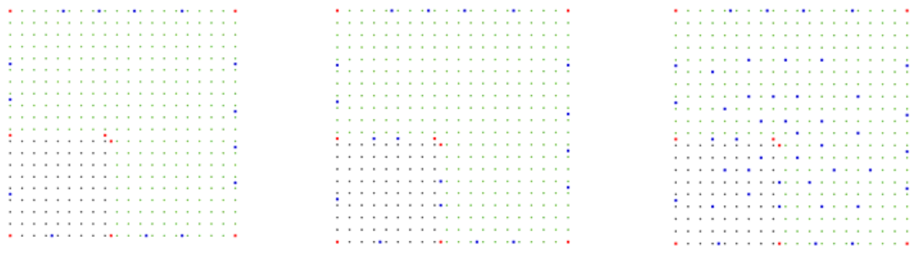


Fig.6 Point Displacement on 2D slices

4. Cross Tissue Boundary Detection

Although the point displacement can generate a well-shaped tetrahedral mesh, there are still some elements which could across boundary of tissues. To solve this problem, we should give a criterion to check whether an element across boundary and present a remesh algorithm to restore the boundary. Fig 8 gives an example of the crossing boundary problem.

4.1 Criterion for crossing boundary

We can separate all the mesh vertices into two groups by their coordinates:

1. Boundary Points (BP): in the coordinates, there is a value which has .5 after the decimal point.
2. Voxel Points (VP): All the coordinates has integer values.

So label each vertex by BP and VP, we can category all the edges in the mesh into 3 groups:

- VP-VP: in this case, we can simply check the tissue type on the voxel point from the segmented volume to find out if the edge cross boundary.
- BP-VP & BP-BP: The crossing status can be determined by the configuration of iso-surfaces in the local neighbourhood, normally one or two voxels.

4.2 Remesh to restore the tissue boundary

The remesh process is just like the flip based tetrahedralization method except that most of time it includes more than 3 tetrahedra. It takes the following three steps:

- Finding all the elements containing the crossing edge
- Finding all the faces and points left to form new elements
- Tessellation and construct new faces and elements

But in fact the above method is invalid when the polygon left for tessellation is not convex. The tessellation is not so simple as to link all other nodes to the start point.

To solve this problem, we present three basic flip operations: flip23, flip32, flip4diagonal (Fig 7).

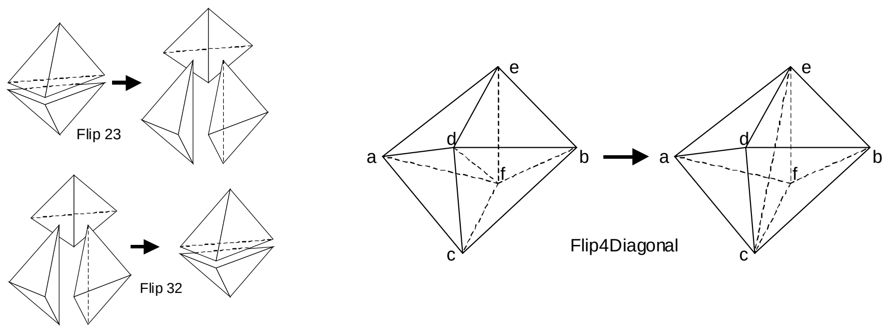


Fig 7 Basic flip operations to restore tissue boundary

The flip operation is depended on the number of the around elements:

- When the number of elements is 3, use flip32 to delete the crossing edge.
- When the number of elements is 4, use Flip4Diagonal to swap the diagonal crossing edge.
- When the number of elements is more than 4, recursively use flip23 and flip4 operations on the around elements to decrease the number to 4 or 3, then use above basic operations to solve the problem.

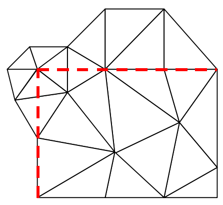


Fig 8. Elements across tissue boundary

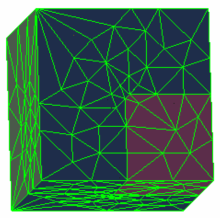


Fig 9. The final tetrahedral mesh

Fig.8 shows the crossing boundary edge on the surface of the sample volume data. Fig. 9 gives the final result with the tissue boundary restored.

4.3 Set element tissue type

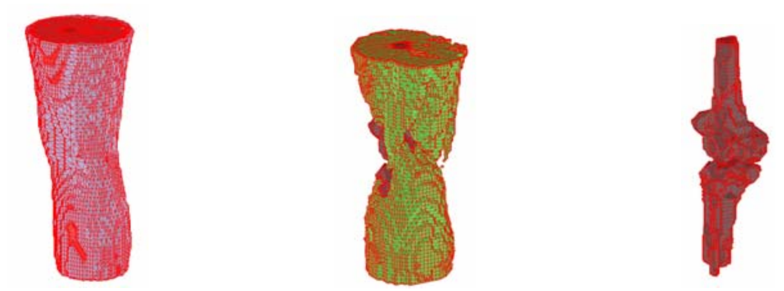
When there is no crossing boundary element, each element's tissue type can be determined by the following rules:

- When there is at least one voxel node in the element, the tissue type will be same as the corresponding volume voxel node.
- When all the nodes of the element are on the tissue boundary, using trilinear

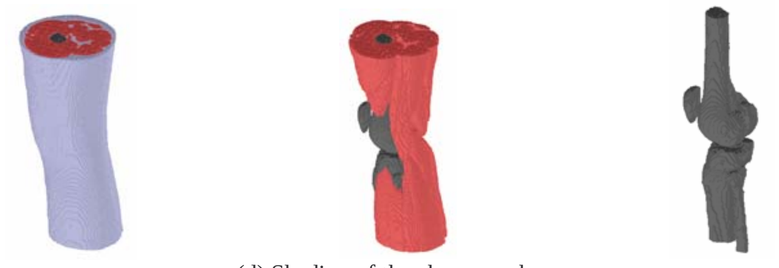
interpolation to compute the exact tissue type at the center of the element.

5. Results

We have implemented the algorithm in our knee arthroscopy simulation system. For a segmented volume with resolution of 297 x 341 x 180, and in total 18,229,860 voxels, our algorithm generates a tetrahedral mesh with 94,953 nodes and 490,409 elements (Fig.10).



(a) Fat + Muscle + Bone (b) Muscle + Bone (c) Bone



(d) Shading of the above meshes
Fig 10 Tetrahedral mesh for the Knee arthroscopy simulation

6. Conclusion

A new tetrahedralization method is proposed to automatically generate three-dimensional finite element meshes for the segmented volume. It requires no user intervention and has the ability to vary the resolution throughout the domain. The validity of the algorithm has been demonstrated via numerous anatomically accurate finite element tetrahedral models produced.

References

1. Nielson, G.M., Junwon Sung, Interval Volume Tetrahedrizatin, Visualization '97,221 -228
2. John M. Sullivan, Jr., Ziji Wu, and Anand Kulkarni, Three-Dimensional Finite-Element Mesh Generation of Anatomically Accurate Organs Using Surface Geometries Created From the Visible Human Dataset, The Third Visible Human Project Conference, October 5 & 6, 2000