A Tetrahedral Model to represent the Left Ventricle Volume of the Heart

Lyudmila Rodríguez, Isabel Navazo, Álvar Vinacua [1]
Email: {lyudmila, isabel, alvar}@lsi.upc.es

Universitat Politècnica de Catalunya
Departament de Llenguatges i Sistemes Informatics
Barcelona, España
Junio, 2002

[1]This work was partially supported by project TIC-2000-1009 funded by the Spanish Ministry of Science and Technology

Contents

Abstract

This report describes an algorithm for the generation of unstructured tetrahedral meshes from two triangular meshes: the internal and external walls of an object. The tetrahedral mesh thus obtained represents the volume between the two given triangular meshes. We choose tetrahedral meshes to represent the volume because they provide more topological flexibility. The algorithm designed for this purpose is based on "advancing front" techniques, but unlike most existing methods based on this concept, our algorithm uses adjacency information about the elements of the volumetric mesh to construct a tetrahedral element. In this manner, it is possible to select, at each step, the neighboring point that guarantees the best tetrahedron, based on an objective function. In addition, the proposed algorithm allows us to have, within the same model, morphological and functional information of interest. This information is very useful, for instance, in the reconstruction of the left ventricle (LV) of the heart, as will be discussed in the paper.

1 Introduction

Over the past several years, the field of medical imaging has grown significantly. Among many other uses and applications, medical imaging assists clinicians in diagnostic, surgical training, and therapy planning tasks. Of particular interest is the clinical diagnosis of cardiovascular diseases, since collectively they represent one of the main causes of morbidity and mortality.

A principal cause of cardiovascular disease is myocardial infarction. It is produced by a progressive narrowing of the coronary arteries (atherosclerosis). If the blockage is severe enough to restrict blood flow to the heart, then the affected heart muscle (myocardium) suffers from lack of oxygen and becomes ischemic. The patient may suffer then a possibly fatal myocardial infarction.

In order to hopefully contribute toward a better clinical diagnosis, we have undertaken a project whose aim is to construct a volumetric model of the left ventricle of the heart, which is the ventricle of primary clinical interest in most cases. Our goal is to assess the location, extent, and severity of possibly hypoperfused regions [5, 8]. The myocardial perfusion distribution is obtained from SPECT (acronym of Single Photon Emission Computed Tomography) imagery. The volume is based on tetrahedra since these geometrical elements can be used to approximate three-dimensional sets. Additionally, tetrahedral volume meshes provide more topological flexibility, and are very useful to represent deformation, rendering and surgical cuts of soft tissue structures [4, 9].

The algorithm is based on advancing front techniques, but includes additional, innovative features. First, it works with two disconnected triangular meshes, that represent the endocardial (internal) and the epicardial (external) surface of the heart. Second, the algorithm obtains the best tetrahedron in each step, producing well shaped elements. Third, it tries to minimize the number of tetrahedral elements, preserving the two original surfaces. Subsequently, we use the perfusion information to subdivide a tetrahedral element in cases where the perfusion inside the tetrahedron is not homogeneous enough.

Our approach differs from the previous work in different ways. We use the adjacency information between the previously constructed elements to generate a new tetrahedron. This allows us to decrease the number of nodes (vertices) that have to be verified to form the set of candidates nodes. The initial front is formed by three faces that are obtained as a result of joining an external face with one internal node. The amount of intersections are reduced due to the reduction in the front size relative to other methods.

An overview of previous work is discussed in section 2. Our tetrahedral mesh generation algorithm is presented in section 3. Results are shown in section 4, and conclusion and future work in section 5.

2 Previous work

Several approaches have been developed to generate an unstructured 3D mesh and particulary a tetrahedral mesh. These methods include Delaunay 3D triangulation [10, 12, 15, 23], octrees [2, 3, 21], advancing front techniques [6, 14, 15, 19, 20] and a particular proposal based on planar cross sections [1].

Lo [15] comments the difficulties involved in 3D Delaunay triangulation as a finite element mesh generator. These include the existence of degenerate cases (when a newly inserted node lies on the surface of a circumsphere associated with some existing tetrahedron), creation of slivers (tetrahedra with near zero volume) and also tetrahedra intersecting the surface mesh in non-convex domains. Weatherill and Hassan [23] attempt to solve the last problem subdividing the tetrahedra that cut the surface mesh. Krysl and Ortiz [12] proposed a method that, from a boundary triangulation, finds the decomposition of its volume into tetrahedral elements such that the triangulated boundary surfaces are 'reasonably' approximated by a collection of tetrahedral faces. The main drawback of their proposal, aside the modification of the boundary mesh, is the necessary post-processing step to reduce the number of the slivers in the mesh.

The octree method encloses the model in a bounding box. Then, this box is split into eight congruent cubes, each of which is split recursively until a desired criterion is satisfied. Shepard and his collaborators have developed several octree-based mesh generators for polyhedral domains [3, 21]. Their original generator tetrahedralizes leaf cubes using a collection of predefined patterns. They kept the pattern numbers down by the assumption that each cube is cut by at most three faces. This algorithm has proved to be useful. Its main weakness is that it tends to be rather complicated.

Bajaj [1] and his group, propose an alternative method to generate a tetrahedral mesh from planar contours. Their algorithm uses a multi-pass tiling approach. Given the solid bounded by two adjacent contours and surface triangular meshes, the algorithm tetrahedralizes that region with the additional constraint of a pre-triangulated top face. The first drawback of their proposal is the necessity to have a good surface triangular mesh algorithm from planar cross sections to resolve the correspondence, tiling and branching problem [16]. Additionally, fully automated image segmentation is an unsolved problem [18]. Particulary, it is hardest in heart images, where the LV boundary may not consist of strong features (it has a low gradient

magnitude).

As already stated, our proposal is based instead on the advancing front technique, which will be explained in detail in the next section.

2.1 Advancing Front Method

Before we explain our approach, we are going to illustrate the process of any advancing front method.

The first stage of the volume tetrahedrization process involves the triangulation of the domain bounding surfaces. Tetrahedrization of the domain volume is based on these triangulated surfaces. A bad triangular mesh quality can produce tetrahedral meshes with poor shape elements. Once the triangular facets have been consistently calculated, the advancing front methods follow the next steps:

1. Form the initial front with the collection of triangles of the domain boundary.
2. While the front is not empty, apply:
   (a) Select a triangular face from the front (active face).
   (b) Create a candidate nodes set (feasible vertices of the new tetrahedra).
   (c) From the set of candidate nodes, choose the node that produces the best tetrahedron when combined with the active face.
   (d) Update the front.

We are going to explain these steps, to emphasize the differences with our proposal technique.

The quality of the resulting volume mesh depends on the surface mesh quality, therefore most of these algorithms need a good domain triangulation. Different methods are used to control the sizes and directions of formed mesh elements.

The method for selecting the triangular face changes in different implementations: the first/last front face, depending on the metric of the tetrahedron to be formed, etc.

A node is candidate if the proposed tetrahedron lies in the interior region and does not cut across the front. These nodes can be nodal points (vertices of the front), or interior or new nodes. Some implementations create additional interior points (called Steiner points) before forming the initial front. In other proposals, these nodes are created as needed.

The selection of a node to form a tetrahedral element is based on different shape measures. It can be shape quality, size, minimum solid angle, dihedral angles, etc. [7, 13].

3 Our proposal

We have already stated that our approach has been created to reconstruct the myocardial volume. As input we have two disconnected surfaces, the epicardial and endocardial surfaces of the heart left ventricle, that we will call external and internal surface respectively for short. The internal surface is inside the volume bound by the external surface. In our method, instead of building the initial front with the collection of the domain boundary triangles, we form it with three new faces. These faces are the internal faces of a tetrahedron formed by an external triangular face and one internal node. Particulary, we choose the node on the internal surface that produces the best tetrahedron. The three new faces, named interior faces, form the initial front.

Our method exploits the adjacency information between the tetrahedral elements. For each front face it is possible to find a set of candidate-nodes taking only into account its adjacent faces, either on the boundary or on the front. This strategy accelerates searching for candidate nodes, because the algorithm avoids a lot of restriction verification. Remember that the traditional algorithms create a candidate nodes set from vertices of the front, interior nodes or new ones, and a node is a candidate if the proposed tetrahedron lies in the interior part and does not cut across the generation front.

Other important difference with the previous methods is that our algorithm only adds Steiner points when either the volume mesh has gaps (untetrahedralizable regions) or the variance of the perfusion values at a tetrahedron's vertices is superior to a threshold value. We do this to avoid increasing the mesh's density. Furthermore it is desirable to have a fast computation of the underlying deformation to support interactive simulations. In the current implementation, these two cases are treated in a postprocessing step.

The first case, that of untetrahedralizable regions, corresponds to Schönhardt prisms. Figure 1 shows their simplest form. Our algorithm resolves this in two different ways: if one side of the prism has two adjacent coplanar faces belonging to the domain boundary, the problem is corrected doing an edge flip. This yields three tetrahedra (figure 2). If that is not the case, the algorithm adds a Steiner point inside the prism. This yields eight tetrahedra (figure 3).

The second case requiring postprocessing, that where the perfusion values at the vertices of a tetrahedron are too dissimilar, is addressed by subdi-

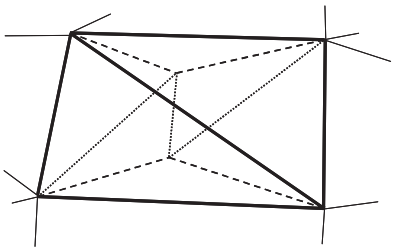

Figure 1: Untetrahedralizable regions: Schönhardt prism

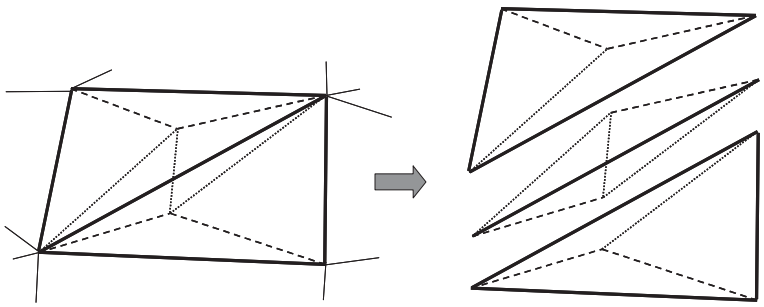

Figure 2: Untetrahedralizable regions are corrected doing an edge flip, if a prism wall bellows to domain boundary

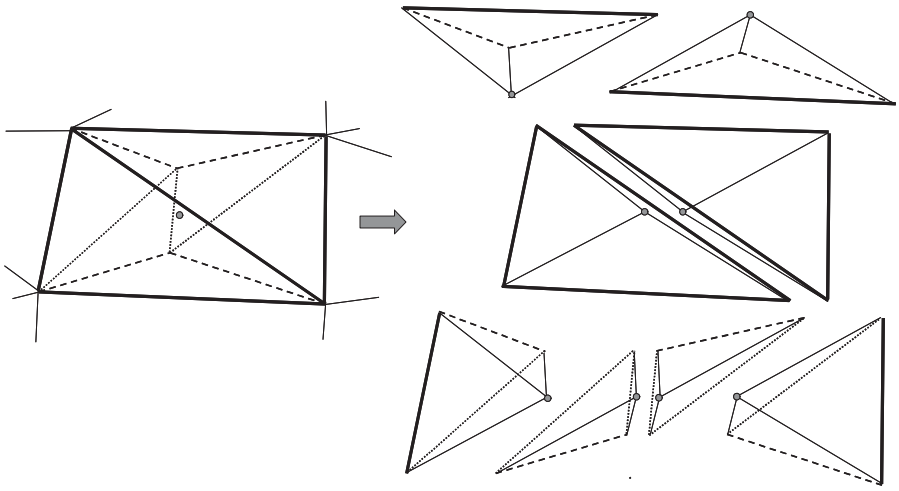

Figure 3: Untetrahedralizable regions are corrected adding a Steiner point

viding the tetrahedron element and propagating the changes to neighboring tetrahedra. At the moment, we are working on the optimal solution to this problem.

Our proposal follows these steps:

1. Form an initial tetrahedron with brute force (i.e. by exhaustive search of the best vertex associated with a randomly chosen triangle of the external surface.)
2. Add the three new faces to the front
3. While the front is not empty, apply:
   (a) Select a triangular face from the front
   (b) Create a candidate nodes set. These nodes are the suitable points adjacent to the active face (see section 3.1).
   (c) If the candidate nodes set is not empty
      i. Choose the node with best quality value (see section 3.2).
      ii. Update the front.
   (d) If candidate nodes set is empty
      i. Add the active face to an auxiliary front (see section 3.3).
      ii. Extract the active face from the front.
4. While the auxiliary front is not empty:
   (a) Select a triangular face from the auxiliary front
   (b) Find the adjacent faces from the auxiliary front.
   (c) For each adjacent face, go to 4b
   (d) When the collected faces form a prism:
      i. If two adjacent faces belong to the domain boundary and are coplanar, do an edge flip of these faces.
      ii. Else add a Steiner point inside the prism.
      iii. Apply step 3

3.1 Creation of a candidate nodes set

To create a candidate nodes set for an active face, we use the adjacency information. For each vertex of the active face, we take its skirt and consider its vertices as aspirants do join the candidate nodes set. Several restrictions must be checked at each vertex before adding it to the nodes set, in order to invalidate the nodes that would not constitute a valid tetrahedron vertex with those of the active face.

The first of these restrictions consists in verifying that the vertex is not behind the active face. We reject the vertex if the dot product between the active face's normal *vNormal* and the vector from the vertex to the centroid of the active face, called *vPoint*, is less than zero ($vNormal \cdot vPoint < 0$). The vertices of the triangular faces are ordered so that their normals always point outwards (that is in the opposite to the direction of advancement of the front) [14, 6]. The figure 4 shows the active face's normal (gray face) and two possibles candidate nodes. After it is calculated the dot product between *vNormal* and $vPoint_i$, $i = 1..2$, $vPoint_2$ is discarded (white node). At right, it can be seen the tetrahedron formed with the active face and the candidate node (black node).

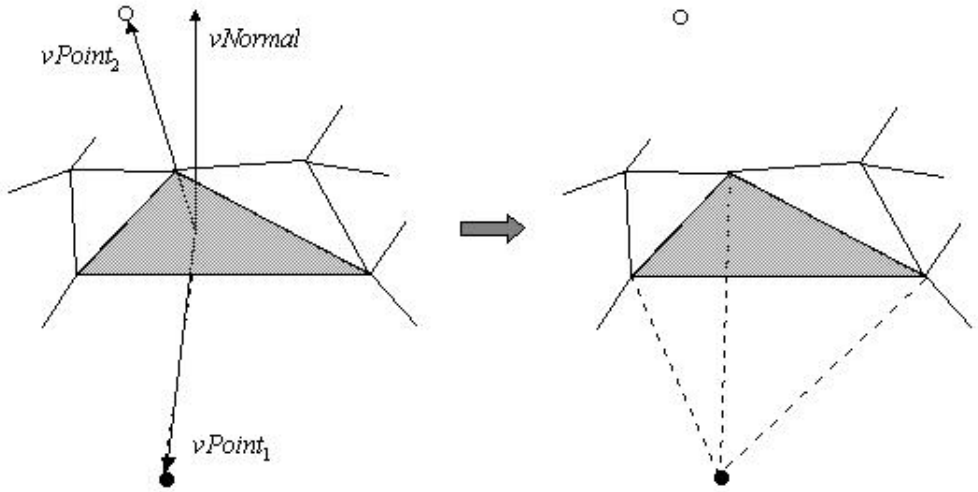

Figure 4: A vertex is candidate if $vNormal \cdot vPoint < 0$

A second restriction discards vertices that can produce invalid faces. We examine each of the faces that would be formed by an edge of the active face and the candidate node. If all three vertices lie on one of the input triangulations but are not the vertices of a triangle in that triangulation, the resulting face would be invalid and the node is discarded, as can be seen in the figure 5. At left, it is shown an initial configuration to the active face painted in gray. Nodes discardes due the normal restriction are painted in white and candidate nodes are painted in black, like we seen above. The gray nodes corresponds to vertices which formed invalid faces, as is shown in the image at right.

A third restriction discards vertices behind adjacent faces. The adjacent faces that are in the forward hemispace of the active face are called clipping planes (CP). Hence, the CPs fulfill: $vNormal \cdot vAdjacent < 0$, where *vNormal* is the active face's normal, and *vAdjacent* is the vector between the

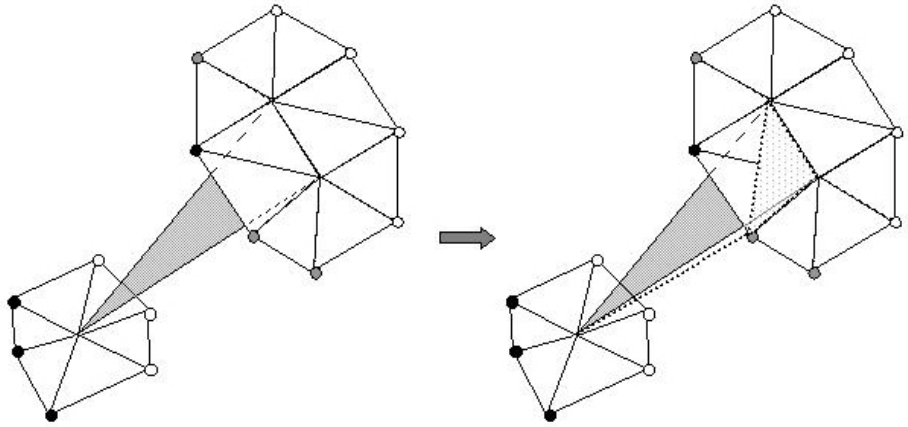

Figure 5: A vertex is discard if produce an invalid face

active face's centroid and the adjacent face's centroid. If the active face has any CPs defined, the algorithm verifies for each vertex in the nodes set if it is ahead of each CP. If the vertex does not satisfy this condition, it is discarded. In other words, the restriction imposed is that $vNormalCP \cdot vPoint < 0$, where $vNormalCP$ is the CP normal and $vPoint$ is the vector from the centroid of the CP to the vertex, as can be seen in figure 6. The active face and candidate nodes are painted in gray. The clipping plane is the white face, neighbor to the active face. The white point cannot select, due is behind the active face.

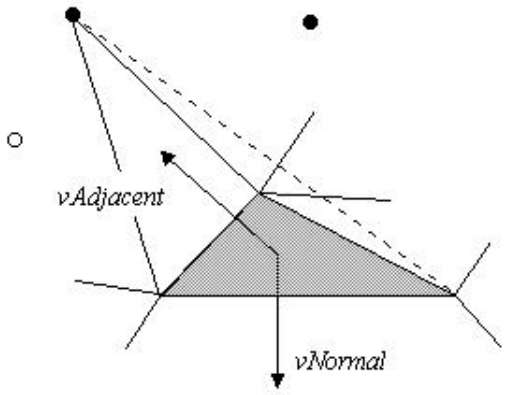

Figure 6: A clipping plane is formed if exist any vector between the active face and adjacent points that satisfy: $vNormal \cdot vAdjacent < 0$

Figure 7 shows an extreme case where all candidate nodes are discarded because each one is behind a CP. The candidate node of the face A is behind the plane Π_C, the plane where the face C lies. The face's B vertex is behind the plane Π_A. Finally, the vertex of the face C is occluded by the plane Π_B. This case, a Schönhardt prism, is resolved adding a Steiner point.

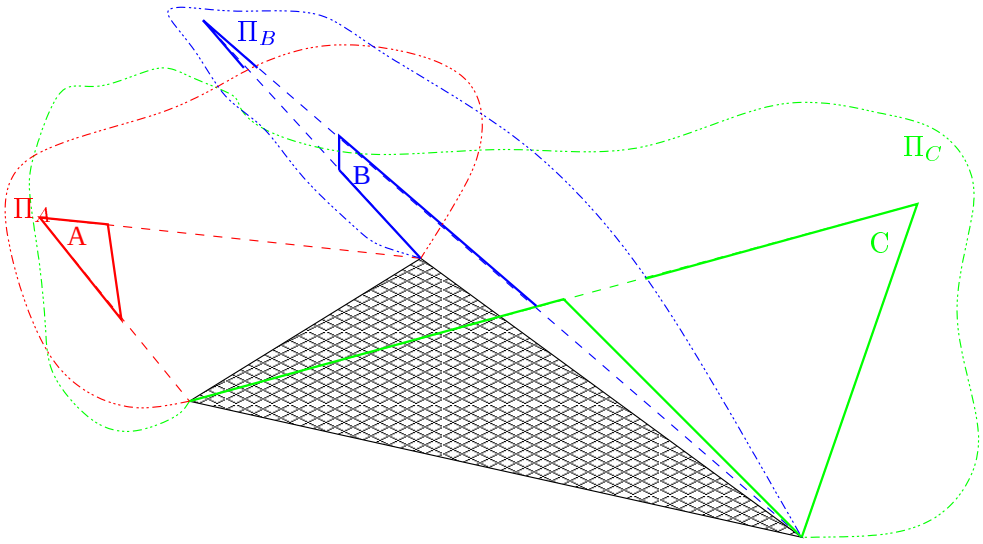

Figure 7: All candidate nodes are discarded because each one is behind a CP

The last restriction avoids intersections with already formed tetrahedra. Intersections occur if one face of the potential tetrahedron cuts or overlaps an already formed tetrahedron. To verify this, we apply Möller's algorithm [17] adapted to our problem to control overlapping. The modifications are based on the detection of the intersection of two lines in 3D [11]. Intersections are verified among the new faces and the faces belonging to the front. Additionally, the boundary domain faces that overlap the bounding box of the potential tetrahedron are also tested. Finding the boundary domain faces is not costly, for these faces are stored in our volume data set.

3.2 Choose a candidate node

The result of the algorithm described in the previous section, is a candidate nodes set. Of all these vertices, we choose the one that produces the best quality tetrahedron. There exist different tetrahedron shape quality measures. These measures are continuous functions that evaluate the quality of a tetrahedron [7, 13]. In particular, we combine the Radius Ratio with the distance between the vertex and the active face because we are interested in choosing the closest node to the active face which produces the best shaped

tetrahedron. Our objective function is then given by

$$quality_i = \frac{DistPlano(vertex_i)}{RadiusRatio(vertex_i)^2}$$

where $i = 0..number\ of\ candidate\ nodes$, $DistPlane(vertex_i)$ is the distance between the candidate node i and the plane defined by the active face, $RadiusRatio(vertex_i)$ is the ratio of *inradius* to *circumradius* scaled by 3 of the tetrahedron formed by the active face and the candidate node i [13]. The $RadiusRatio \leq 1$ for any tetrahedron, and the equality holds iff the tetrahedron is regular.

This quality measure yields the best results among those we have studied so far. More work is desirable in this area, as mentioned later in section 5.

We now compute the quality of all candidate nodes that satisfy all our restrictions and choose the best one (i.e. the one with smallest value of the quality measure) as the new vertex of a tetrahedron. The figure 8 shows at right two differents tetrahedra to the based configuration (left). In the same way as above, the active face is painted in gray and the candidate nodes are painted in black.

3.3 The auxiliary front

When the algorithm can not generate a tetrahedral element on the active face, the active face is extracted from the front and is added to an auxiliary front. This auxiliary front has all the faces which the algorithm could not construct a tetrahedron for, and correspond to Schönhardt prisms.

In our present implementation, these cases are treated in a postprocess, but during the construction of the tetrahedral mesh we are able to detect this problem. We are studying ways to deal with this problem as soon as it is detected, so we can do without the postprocessing of our mesh.

4 Results

We have run several simulations with SPECT data. This data are obtained by injecting a radioactive marker to the patient. The marker is known to bind to specific tissue, depending on the purpose of the test. For cardiac ailments, markers that bind to muscle are used, so large counts correspond to well irrigated muscle, and therefore show the perfusion of blood throughout the myocardium. The images obtained, however, are of a low resolution and somewhat blurred. Blurring comes from the random nature of radioactive decay, but even more from the fact that we are not able to take an instantaneous picture. In fact the data for these images are gathered through a long

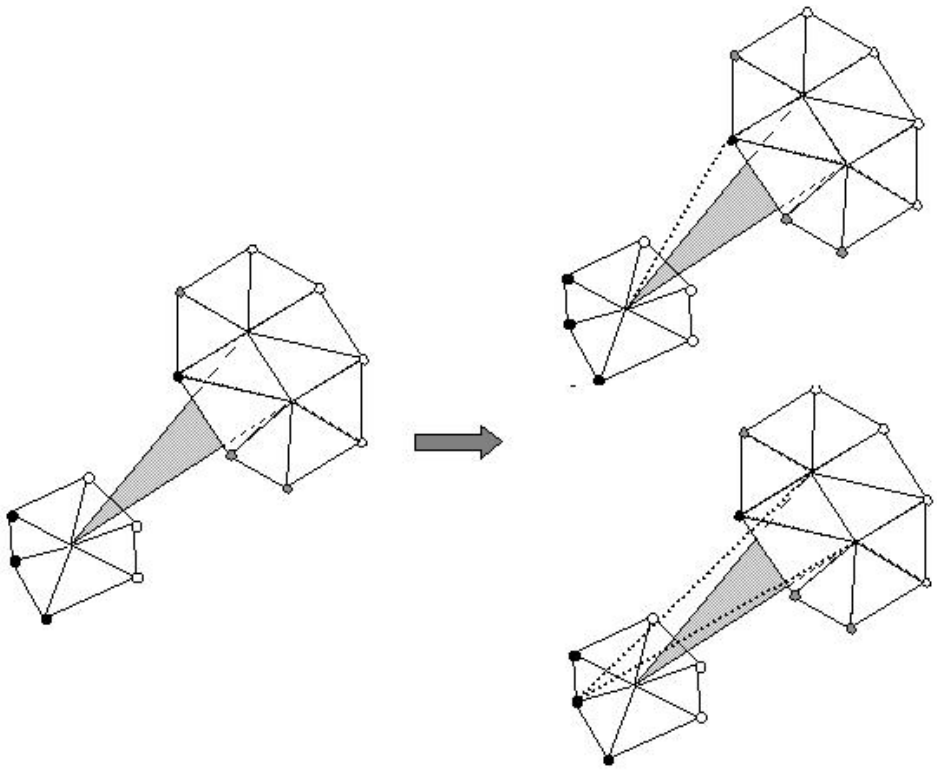

Figure 8: It is shown two differents tetrahedralization to the active face in gray

process (several minutes) during which, of course, the heart is not still. The effect of the heart's beating is minimized by synchronizing the data capture with the heart through an electrocardiogram.

To segment the data and construct the bounding surfaces, we have used techniques based on deformable models (see [22]).

Figure 9 shows the starting data. The color palette is chosen so that red zones correspond to high perfusion. Figure 10 shows a volume rendering of the grid using 3D textures.

Table 1 summarizes the results for three representative test cases: One data capture run on a Mayo Clinic Phantom, and two synthetic data sets constructed modelling the two surfaces with quadrics. These data sets have been chosen because they represent more faithfully the features of the algorithm. When dealing with real data from patients, the reconstruction techniques become dominant upon the quality of the solution, for not only is the data more noisy, but whole parts of the surface are missing, since ischemic regions do not produce any counts at all (and therefore appear

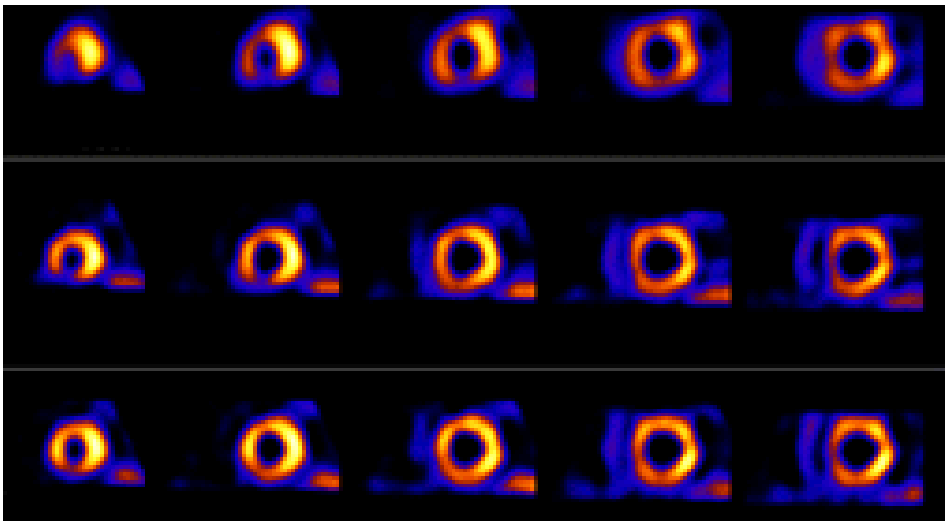

Figure 9: SPECT images

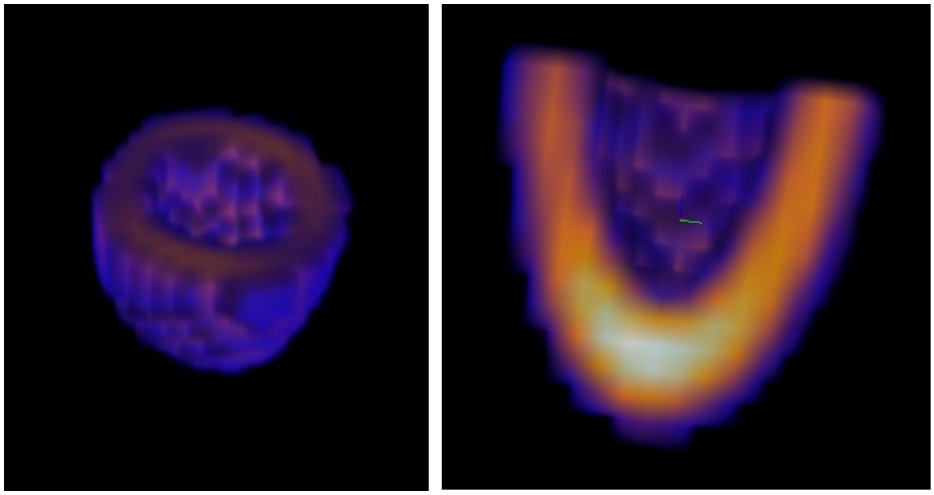

Figure 10: Volume rendering using 3D textures and a cut with a plane

completely black in the captured data). Dealing with that kind of data is a big problem in itself, which we shall address in a separate paper.

For each of the chosen datasets, we list the number of faces in each of the triangular meshes, and the number of tetrahedra used by our algorithm. As a measure of the quality of the resulting tetrahedrization, we list here the quotient of the radii of the circumsphere and the insphere divided by three. This measure is adimensional, and allows a fair comparison of how the algorithm has performed across different models. Note that this is not

Table 1: Results.

	#F.E.S.	#F.I.S.	#Tetra	Quality
Phantom	560	140	966	0.469885
Quadric 1	48	30	118	0.653416
Quadric 2	986	270	2445	0.446748

the measure that our algorithm uses, which contains distances and is therefore model dependant. This is not a problem since we use it to compare tetrahedra of the same model, but its values would not be meaningful when tabulated. The quality function that we use in our algorithm —as described in section 3.2— is however the one that yields the best experimental results.

The table displays an acceptable average quality of the resulting tetrahedra over meshes of different sizes. A study of the histogram of resulting qualities shows that there are about 10% of the tetrahedra which have poor quality, and correspond to the apex of the heart. The number of tetrahedra filled in during the postprocessing step is less than 5% of the total.

Finally, figure 11 shows steps in the construction of a tetrahedrization. The active face appears red. Green dots correspond to candidate nodes. The exterior mesh is shown in gray wireframe, and the interior in red. The figure shows four steps in the construction of the tetrahedrization.

5 Conclusions and Future Work

We have presented an efficient, specialized algorithm to tetrahedrize the myocardium from SPECT data. This technique may also be of value in the tetrahedization of others volumes bound by two surfaces.

We are presently working on several improvements and fine-tunings of the algorithm, including: the subdivision of exceedingly non-homogeneous tetrahedra, avoiding our postprocessing by solving Schönhardt's configurations on-line and testing other quality measures, especially adimensional measures that yield scale invariance.

Our plans for future work include:

- Time dependent tetrahedrizations of the full cardiac cycle from gated-SPECT data.
- Support for interactive deformations and changes in topology (cuts).

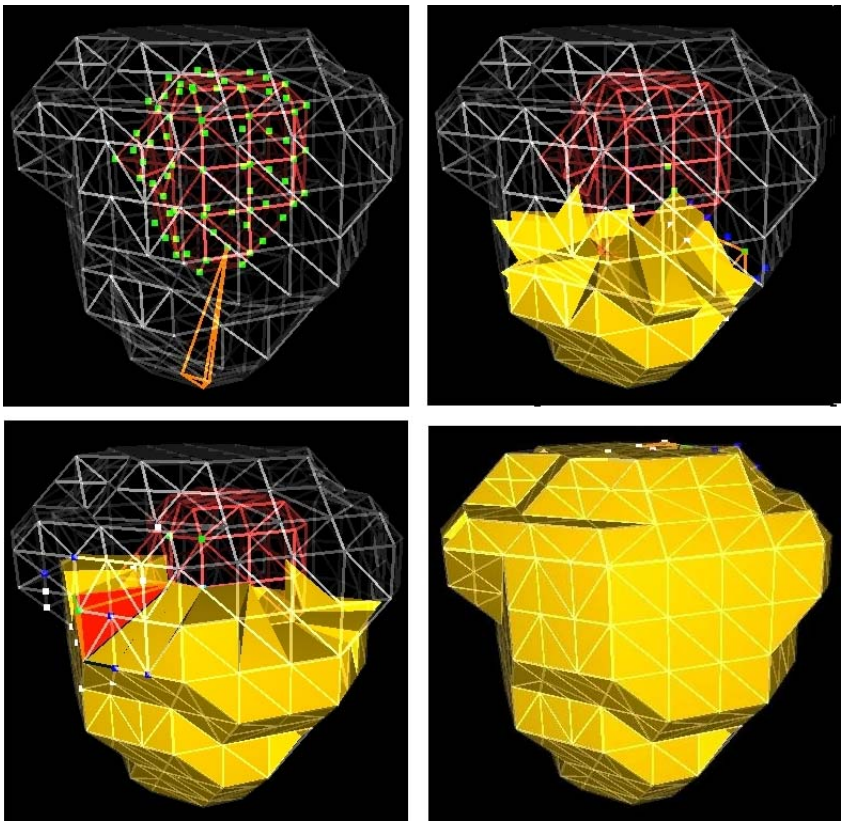

Figure 11: It shows four steps of tetrahedrization process

Bibliography

[1] C. Bajaj, E. Coyle, and K. Lin. Tetrahedral Meshes from Planar Cross Sections. *Computers Methods in Applied Mechanics and Engineering*, 179(1):31–52, 1999.

[2] M. Bern and D. Eppstein. Mesh Generation and Optimal Triangulation. In Ding-Zhu Du and Frank Hwang, editors, *Computing in Euclidean Geometry*, volume 1 of *Lecture Notes Series on Computing*, pages 23–90. World Scientific, 1992.

[3] M. Bern and P. Plassmann. Mesh generation. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, chapter 6. Elsevier Science, 1999.

[4] D. Bielser and M. Gross. Interactive Simulation of Surgical Cuts. In *Proc. of Pacific Graphics*, pages 116–125. IEEE Computer Society Press, 2000.

[5] J. Candell, J. Castell, and S. Aguadé. *Miocardio en riesgo y miocardio viable: Diagnóstico mediante SPECT*. Editorial Doyma, 1998.

[6] C. Chan and K. Anastasiou. An automatic tetrahedral mesh generation scheme by the advancing front method. *Communications in Numerical Methods in Engineering*, 13:33–46, 1997.

[7] J. Dompierre, P. Labbé, F. Guibault, and R. Camarero. Proposal of Benchmarks for 3D Unstructured Tetrahedral Mesh Optimization. Technical Report CERCA R98-91, Centre de Recherche en Calcul Appliqué, École Polytechnique Montréal, 1998.

[8] A. Frangi, W. Niessen, and M. Viergever. Three-Dimensional Modeling for Functional Analysis of Cardiac Images: A Review. *IEEE Transactions on Medical Imaging*, 20(1):1–26, 2001.

[9] F. Ganovelli, P. Cignoni, C. Montani, and R. Scopigno. A Multiresolution Model for Soft Objects supporting interactive cuts and lacerations. *Computer Graphics Forum*, 19(3):271–282, 2000.

[10] P. L. George. Tet meshing: construction, optimization and adaption. In *Proc. 8th International Meshing Roundtable*, pages 133–141, USA, 1999.

[11] R. Goldman. Intersection of two lines in three-space. In Andrew Glassner, editor, *Graphics Gems*, page 304. Academic Press Profesional, 1990.

[12] P. Krysl and M. Ortiz. Variational Delaunay approach to the generation of tetrahedral finite element meshes. *International Journal for Numerical Methods in Engineering*, 50:1681–1700, 2001.

[13] A. Liu and B. Joe. Relantionship between tetrahedron shape measures. *BIT*, 34:268–287, 1994.

[14] S. H. Lo. Volume Discretization into Tetrahedra-I. Verification and triangulation of boundary surfaces. *Computers & Structures*, 39(5):493–500, 1991.

[15] S. H. Lo. Volume Discretization into Tetrahedra-II. 3D triangulation by Advancing Front approach. *Computers & Structures*, 39(5):501–511, 1991.

[16] D. Meyers, S. Skinner, and K. Sloan. Surfaces from contours. *ACM Transactions on Graphics*, 11(3):228–258, 1992.

[17] T. Möller and B. Trumbore. Fast, minimun storage ray/triangle intersection. *Journal of Graphics Tools*, 2(1):21–28, 1997.

[18] E. Mortensen and W. Barrett. Interactive Segmentation with intelligent Scissors. *Graphical Models and Image Processing*, 60(5):349–384, 1998.

[19] A. Rassineaux. Generation and optimizationof tetrahedral meshes by Advancing Front Technique. *International Journal for Numerical Methods in Engineering*, 41:651–674, 1998.

[20] E. Seveno. Towards an adaptive advancing front method. In *Proc. 6th International Meshing Roundtable*, pages 349–360, USA, 1997.

[21] M. Shephard and M. Georges. Three-dimensional mesh generation by finite octree technique. *International Journal for Numerical Methods in Engineering*, 32:709–749, 1991.

[22] A. Susín, I.Navazo, A. Vinacua, and P. Brunet. Dynamic Recognition and Reconstruction of the Human Heart. *15th International Conference on Pattern Recognition*, 4:88–93, 2000.

[23] N. Weatherill and O. Hassan. Efficient three-dimensional Delaunay triangulation with automatic point creation and imposed boundary constraints. *International Journal for Numerical Methods in Engineering*, 37:2005–2039, 1994.