PHILIPS

- Research Master Thesis, Master 2 MVA -

Segmentation of the Left Atrial Appendage in 3D images

Pol Grasland-Mongrain*
under the supervision of Olivier Ecabert†

Philips Forschungslaboratorien Aachen
Weisshausstrasse 2
D 52066 Aachen

20. April 2009 - 28. August 2009

*pol.grasland-mongrain@ens-cachan.fr
†olivier.ecabert@philips.com

Contents

Abstract

The main goal of my master thesis was to segment a part of the heart called *Left Atrial Appendage* (also called *left atrium appendage*, or older *left auricular appendix*), using a method based on the deformation of a mesh with a fixed topology.

The research labs of Philips have created a model-based method to segment most parts of the heart : both atria, both ventricles, and trunks of great vessels. This method is automatic and quite accurate, with a success ratio of about 138 into 140 patients. The Left Atrial Appendage (LAA) is here not segmented and represented like a trunk, due to the difficulty in adapting a mesh with a fixed topology to the highly variable shape of the LAA. This trunk allows the coronary arteries between LAA and myocardium to stay visible. However, the accurate delineation of the LAA is important for the planning of the electrophysiology treatment (e. g. atrial fibrillation).

My work was an extension of the method created by Philips : this method adapts a model to the heart ; once the adaptation is made, the position of the LAA is known, and the segmentation begins. The principle of this segmentation was to create a mesh with an high resolution at the base of the LAA, which inflates until it fits to the exact shape. The deformation is made by minimizing two types of energy, external and internal.

My work can be separated in a few major parts :

- Manual Segmentation of the LAA in 17 patients, to generate groundtruth element and to have a numerical evaluation of the quality of the automatic segmentation.
- Modification of the current Philips model by building a mesh separating LAA and left atrium, and another mesh representing the LAA which will inflate.
- Development of a code which calculates different energies, and inflates the mesh.
- Development of a code to compute histograms of the LAA area, to have an accurate threshold between LAA and other anatomical structures. This threshold is used by one energy during the inflation.
- Qualitative and quantitative evaluation of the results.

1 Introduction

1.1 Motivation

Cardiovascular disease is one of the major cause of deaths in the world. Having a model of the heart of patients can greatly help the physicians to diagnosis and treat disease. For example, the atypical thickness of the myocardium wall or the poor ejection ability of the left ventricle can be the indicator of a disease. The Left Atrial Appendage, subject of my intership, is involved in various heart diseases, including thrombosis building, cardiac fibrillation...

As the heart is continuously moving, it is quite difficult to image. Complementary techniques have been developed, the main ones are UltraSound (US), X-Ray Computed Tomography (CT), Magnetic Resonance Imaging (MRI), Single Photon Emission Computed Tomography (SPECT), and at a less extent Positron emission tomography (PET). Computed Tomography is still widely used: its excellent spatial resolution, ease of use and low cost help to compensate for the use of radioactive beams. Sometimes, only CT can be used: a patient with a pacemaker cannot have an MRI, for example.

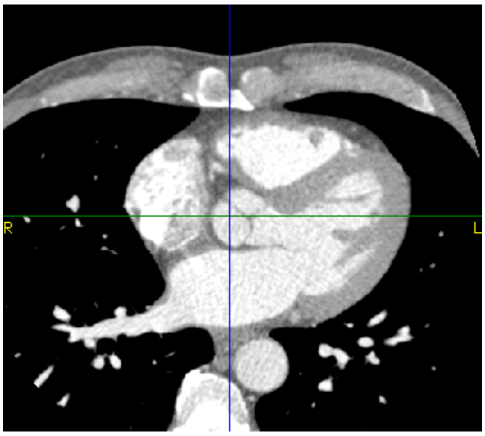


Figure 1: CT axial view, from feet to head, of the heart of a healthy patient

One major goal today is to accurately segment the heart of a patient. Segmentation is the process of assigning labels to region in the image. In a model-based approach, this can be achieved by having a 3D model of the heart of the patient with annotated faces and vertices, for example left or right ventricle, left or right atrium.... Even if the segmentation could be manually done, it is practically almost impossible in a daily work. One CT detection can have thousands of slices, especially if we consider several cardiac phases. Moreover, this can lead to wide variations, even from the same operator. Most of the current techniques use semi-automatic methods, which are a good compromise between time and segmentation efficiency. But recent fully-automatic methods have proven to be accurate and fast enough for clinical use. But these methods can still be improved, and most of these methods are focusing only on the main part of the heart: left and right ventricle, left and right atrium.

Although the Philips method segment a lot of anatomical structures (ventricles, atria, great vessels, pulmonary arteries), the LAA is now not segmented and is thus represented like a trunk. The current model-based method can not be simply applied because the shape of the LAA is too variable. We thus had to think about a new method which could adapt to a variety of shapes and sizes, don't stay stuck in a noisy part or in the contrary "leak" in another organ (a simple region-growing method is consequently not appropriate), and in the same time remains regular enough.

The goal of this master thesis was thus to accurately segment the Left Atrial Appendage (LAA).

1.2 Description of the Left Atrial Appendage

The figure 2 shows the position of the Left Atrial Appendage in the heart.

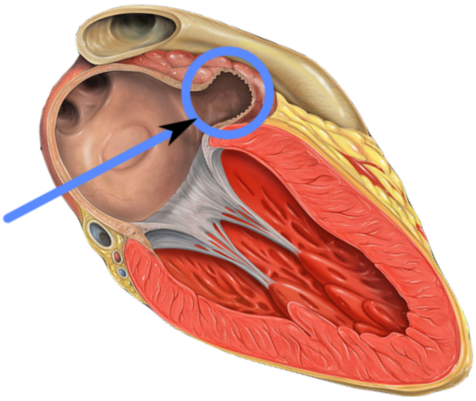


Figure 2: Cut view of heart. The arrow points to the Left Atrial Appendage. (cc) Patrick J. Lynch 2006

The LAA has a "windsock" shape, mostly tubular and hooked. Its size varies from 1 to 19 cm^3 [6]. It can have between 1 and 4 lobes. Figures 3, 4, 5 shows some LAA segmented by hand. It is slightly different from the right atrium appendage which is more massive.

Although the function of the LAA is not perfectly known, most physicians think today that it serves as a reservoir for the atrium, and helps the maintenance and the regulation of the heart function. It has been observed that its elimination causes various heart problem [3].

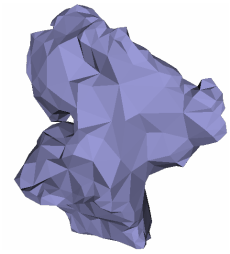


Figure 3: Shape of LAA

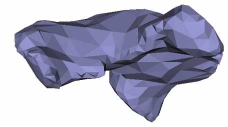


Figure 4: Shape of LAA

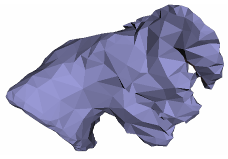


Figure 5: Shape of LAA

The LAA hides the coronary arteries, which have an important role in the heart working; it has sometimes to be surged, e.g. in case of thrombosis. Some physicians wonder if it could be ablated to prevent some complications induced by atrial fibrillation [7]: "*Oral anticoagulation significantly reduces the rate of stroke or embolism in patients with atrial fibrillation. Although oral anticoagulation therapy has been shown to be safe and effective, especially in patients who monitor it by themselves, it is often difficult to achieve a well-controlled therapeutic range of anticoagulation over long periods of time. Contraindications to oral anticoagulation exist in a considerable number of patients. Even in atrial fibrillation patients without contraindications for oral anticoagulation, it is underused due to patient-related, physician-related, and health-care-related barriers. [Less than] 50% of patients with atrial fibrillation receive oral anticoagulants. Because in patients with nonrheumatic, nonvalvular atrial fibrillation[, more than] 90% of the thrombi are located in the LAA, an elimination of the LAA, either by resection or occlusion, seems to be an attractive alternative to oral anticoagulation.*" (abstract from [3])

1.3 Work Environment

My master thesis took place in Philips Forschungslaboratorien Aachen. Philips is an international company which works on many high-technologic subjects, including electronics, home appliance, medical devices... This company is located in many countries in the world. In Germany there are two main research labs with activities in medical imaging: Hamburg and Aachen ; a few teams are working in parallel on different heart subjects. In Aachen, I was in the XIS (X-Ray Imaging System) team, which counts about 30 permanent people, including 3 doctorants. This team works on X-Ray-related subjects: X-Ray generators and detectors, acquisition technology, image analysis tools. The heart modelling project was started in 2001 in Hamburg, and began to be developed in Aachen 3 years later with Jochen Peters, Jürgen Weese and Oliver Ecabert. Currently they are still working on it, with a few other members like Richard Kneser, Carsten Meyer, Helko Lehmann... The heart-modelling software is intented to be given with CT scanners when these ones are sold.

My work had to be included in the main framework of Philips. It provided me so one full 3D model, a source code with a lot of already coded functions, but it added some constrains too (no change of topology of the model allowed for example).

2 Background

Local information alone (e.g. only edge detection, or gray-value information) is not enough efficient for segmenting the heart. Use of global information (e.g. expected shape) are thus necessary.

T. McInerney and D. Terzopoulos made a survey about deformable models in medical image analysis, in 1996 [11]. The heart can be segmented in 2 ways: slice by slice, e.g. with *active contours* ("snake") model[12], then connecting the slices together ; or directly in 3D (deformable model). The first method is less robust, because it can create sharp edges and unconnected regions. Besides, active contours are not always accurate for heart modelization, especially because it doesn't adapt to uncommon shapes (like vessels, which have a long and narrow shape). Deformable models, which are using bottom-up constraints (e.g. boundaries) with a top-down a priori knowledge (relative sizes, positions and orientations of parts of the heart, with expected variations), are consequently widely used for modeling heart.

H. C. van Assen *et al.* explains in [15] a method called 3D Active Shape Model (ASM) driven by Fuzzy Inference. This method can segment semiautomatically cardiac CT and MR volumes, and is still studied now. The authors have applied it mostly on left ventricle segmentation, but it could certainly be extended for other part of the heart.

Y. Zheng *et al.* are describing an efficient method in [19] to modelize the four chambers of the heart (left and right ventricle, left and right atrium). This method is separated in two steps: heart localization, using their own algorithm called marginal space learning and another called steerable features, and boundary delineation. This method is, unlike many others, fully automatic and quite fast. It has been tested on a very large dataset (323 volumes from 137 patients), and has a point-to-mesh error between 1.1 and 1.6 mm, which is in the average quality of the semi-automatic methods, but slightly worse than the current Philips method.

Efficient methods for developing statistical model of heart, with a mean shape and expected variations, have been studied by J. Lötjönen *et al.* in [10]. The main methods are: Principal Component Analysis (PCA), which looks at eigenmodes, where only the most significant modes of variation in shape are kept ; Independant Component Analysis (ICA), described in [4], which tries to separate the independant component. One common example to understand the difference between PCA and ICA is two people speaking in the same time (known as the *coktail party* problem), and the resulting sound is taken by two microphones. The PCA will give the most significant variations in the speech, although the ICA will separate the speakers - it is assumed that their speech are statistically independant. Landmark Probability Distribution (LPD) combined with probabilistic atlas, which gives probability of finding a landmark or a structure at a precise location. It was shown experimentally that other types of parametric transformations combining affine transformations can more effectively describe the inter-patients and inter-cardiac phases variability than PCA [13].

The use of atlases is primordial in this type of work. A. Young and A. Frangi explain in [18] the use of statistics with atlases, with some indications of growing accessible database.

3 Current Method at Philips

3.1 Overview

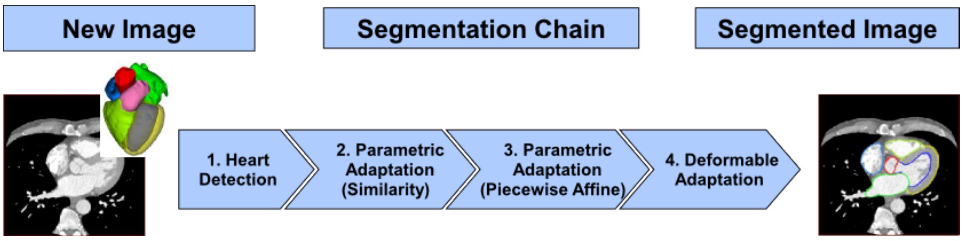


Figure 6: Segmentation chain in the Philips method

The method used in Philips Aachen starts from an already-created model of heart, which is transformed in a few steps, as described in figure 6. This method is fully explained in [14]. For research development purposes, it is included in a software called AdapMeshDemo, where the interface is coded in C#, and the actual algorithm in C++. The figures 7, 8, 9 show a left, middle and right view of the base model of the heart used, with the truncated LAA highlighted in red.

3.2 Heart Detection

The Heart Detection algorithm uses a Generalized Hough Transform. The image is first greatly simplified: only the borders remains, and they are in black and white (no gray value). Then, the model of the heart is translated on each possible position, and counts the number of matching voxels. The higher, the most probable is the position. The model is too scaled at different size for each position, so in the end,

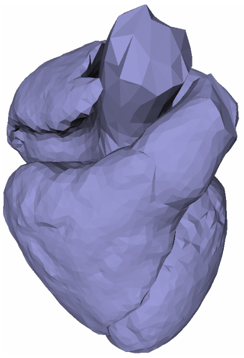


Figure 7: Left view of heart model

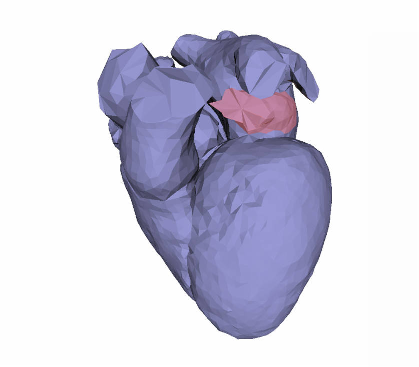


Figure 8: Middle view of heart model

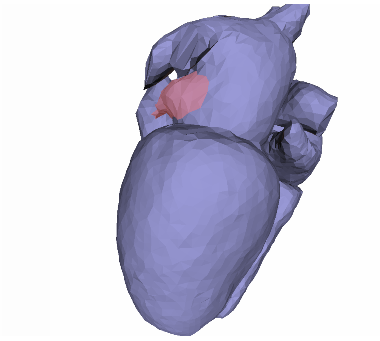


Figure 9: Right view of heart model

the most probable size and position are found. A few improvement are made, like not visiting every voxels, to make the process a little faster. At this step, only a translation and an isotropic scale are computed.

3.3 Parametric Adaptation

The steps 2 and 3 are quite similar: they consists in optimizing a parametric transformation (rotation, translation and scale). The only difference is that in the step 2, the parametric transformation is global: it is computed and applied to the whole heart, while in the step 3, there are several parametric transformations: they are computed and applied independantly to each substructure (left and right ventricle, left and right atrium, etc). In the step 3, the substructures are not disconnected, because the triangles adjacent to the interfaces have a special treatment: the vertices close to the interfaces are moved by the two transformations. These transformations are weighted linearly by their distance to the actual interface; for example, a triangle in one side can be tranformed by 75% of one transformation and 25% of the other.

To find the transformation in each case, the following energy is computed :

$$E_{\text{external}} = \sum_{i=1}^{T} w_i \left(\frac{\boldsymbol{\nabla} I(\boldsymbol{x}_i^{\text{target}})}{\parallel \boldsymbol{\nabla} I(\boldsymbol{x}_i^{\text{target}}) \parallel} \cdot (\boldsymbol{x}_i^{\text{target}} - \boldsymbol{c}_i) \right)^2 \quad (1)$$

where the sum is performed over the mesh triangles.

This energy looks at each triangle centers $\boldsymbol{c}_i$; these triangles centers tends to be attracted towards target points $\boldsymbol{x}_i^{\text{target}}$ detected at the object boundary. One “trick” is used to avoid that the point remains stuck at the target position: the projection of $(\boldsymbol{x}_i^{\text{target}} - \boldsymbol{c}_i)$ onto the normal vector $\boldsymbol{\nabla} I / \parallel \boldsymbol{\nabla} I \parallel$ at the target point makes the energy invariant to movements of the triangle within the object tangent plane. Finally, the weights w_i are large for reliably detected target points and small otherwise.

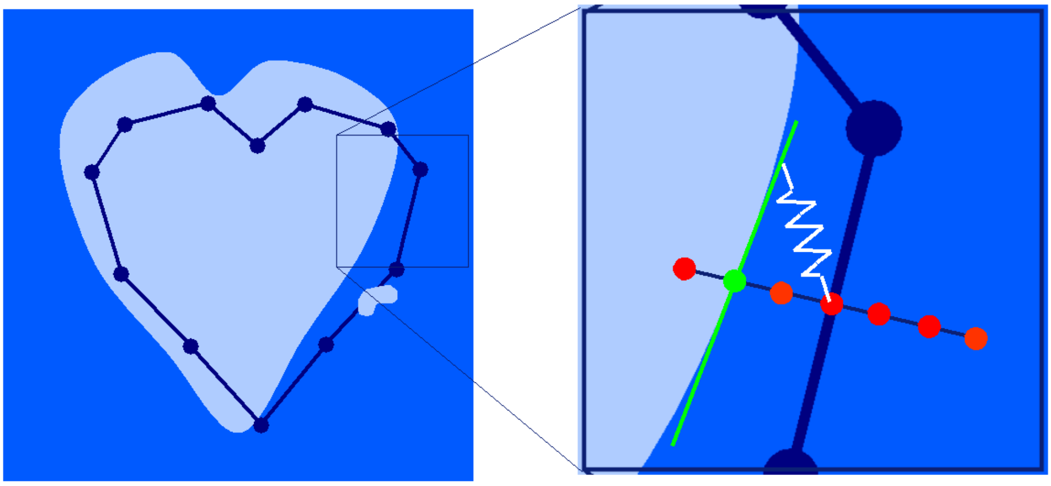


Figure 10: Illustration of the external energy concepts. The target point is searched evaluating the equation (3) along a line perpendicular to the triangle. The point with maximum response is then selected as target point (green point). Finally, the mesh is attracted towards a plane parallel to the boundary at the target point.

The target points are detected in the image maximizing a boundary detection function $F_i(\boldsymbol{n}_i, \boldsymbol{x})$, which is evaluated for each triangle along its normal vector $\boldsymbol{n}_i$. The discrete positions at which the feature function will be evaluated are thus given by

$$\boldsymbol{x}_i^j = \boldsymbol{c}_i + j \cdot \delta \cdot \boldsymbol{n}_i \ , \qquad j = -l, \dots, +l \tag{2}$$

where δ is the sampling distance between the tested positions. The target point is then selected by evaluating the following expression:

$$\boldsymbol{x}_i^{\text{target}} = \arg\max_{\{\boldsymbol{x}_i^j |\, j=-l,\dots,+l\}} \left(F_i(\boldsymbol{n}_i, \boldsymbol{x}_i^j) - D \cdot (\boldsymbol{x}_i^j - \boldsymbol{c}_i)^2 \right) \ . \tag{3}$$

To avoid attraction to too distant edges, a heuristic distance penalty term with weight D is included, which biases the search for nearby target points. Experiments to optimally determine the parameter D were carried out in a comprehensive report [5]. See Fig. 10 for an illustration of the concepts presented above.

The choice of the right feature function is critical for robust segmentation. Basically, the feature function should have a high response at the correct edge and a low response at other locations. One of the most common feature function is the magnitude of the gradient. However, this function is not discriminative between the boundaries of different organs: it just returns a high value at boundaries irrespective of their nature.

Once all the target points are computed, a transformation which minimize the sum of the square of the distance between the centers of triangles and the target point is computed, and applied. It is reminded that the target point can move freely along a plane tangent to the boundary.

3.4 Deformable Adaptation

This last step doesn't constrain the displacement to global or semi-global transformations: each vertex is there allowed to move freely.

The mesh adaptation to a new image is there performed minimizing an energy function, which is made of two contributions. The first one, called *external energy*, is the same as in the section above and attracts the model to the object boundaries in the image, whereas the *internal energy*, penalizes deviations of the deformed model:

$$E = \alpha E_{\text{external}} + E_{\text{internal}}, \tag{4}$$

where α is a parameter which balances the contribution of each energy.

The method about deformable models have already been used successfully in the context of bone segmentation [8], assessment of left ventricular function [17] and radiotherapy planning [9].

3.4.1 Internal Energy

The internal energy will be responsible of (1) maintaining a suitable distribution of the vertices and (2) penalizing deviations between the current state of the mesh and the reference shape model. The first point is addressed considering the difference vectors between the coordinates of two neighboring vertices. The second point is implemented by comparing these vectors with the corresponding ones in the model (see Fig. 11):

$$E_{\text{internal}} = \sum_{i=1}^{V} \sum_{j \in N(i)} ((\boldsymbol{v}_i - \boldsymbol{v}_j) - (\mathcal{T}[\boldsymbol{m}_i] - \mathcal{T}[\boldsymbol{m}_j]))^2 \tag{5}$$

with $N(i)$ the set of indices of the neighbor vertices of vertex $\boldsymbol{v}_i$, and $\boldsymbol{m}_i$ the vertex coordinates of the reference model undergoing a geometric transformation $\mathcal{T}[.]$. This transformation describes allowed global shape variations like e.g. rigid body motion or affine deformations.

3.4.2 Practical Implementation of the Mesh Optimization

Mesh adaptation to a new image is performed minimizing the equation (4). However, this might become a complex nonlinear problem, especially when no assumption is made on the type of global transformation introduced in (5).

In order to make the problem more manageable, we minimize the equation (4) using a two-steps procedure. First, the parameters of the global transformation $\mathcal{T}[.]$ are optimized minimizing the equation (5) with a point-based registration method between the current vertex coordinates $\{\boldsymbol{v}_i\}$ and the corresponding model vertices $\{m_i\}$. In the second step, the vertex coordinates $\{\boldsymbol{v}_i\}$ are updated minimizing (4)

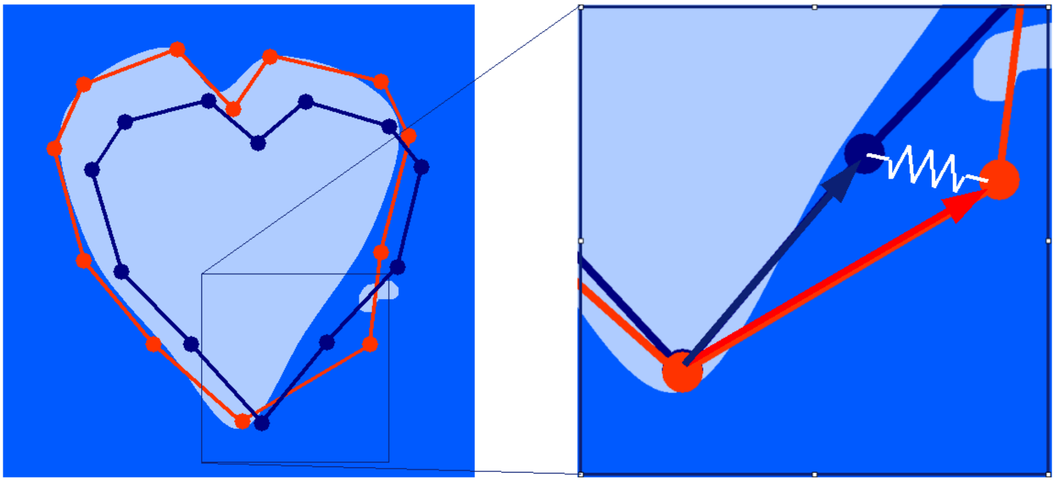


Figure 11: Illustration of the internal energy concepts. The internal energy penalizes deviations between difference vectors in the mesh (red) and PDM (blue).

using the parameters of the global transformation determined in the previous step. Finally, both minimization steps are iterated until convergence.

The equation (4) is quadratic in $\{\boldsymbol{v}_i\}$, meaning that the optimization is equivalent to solving a linear system of equations. Moreover, since only first order neighbours are considered, the linear system of equation is sparse and can be efficiently solved using the conjugate gradient method.

4 Actual Work

We have chosen to treat the LAA differently from the other parts of the heart. Indeed, shape, size and orientation variations are too varying to use the current method described above; and the presence of parts of the heart close to the LAA (myocardium, coronary arteries, left atrium) can create many segmentation errors. The chosen principle was to have a base mesh which "inflates" through the LAA, without changing the topology.

4.1 Implementation in the Philips Framework

The algorithm of the segmentation of the LAA had to be implement in the Philips algorithm. It has been made as a step done after the adaptation of the main model of the heart (see figure 12). This had a few implications :

- The topology of the model cannot change (except in a special circumstance explained the section 4.5)

- Each energy has to be written in a quadratic form: "$(A\mathbf{v} + \mathbf{b})^2$"

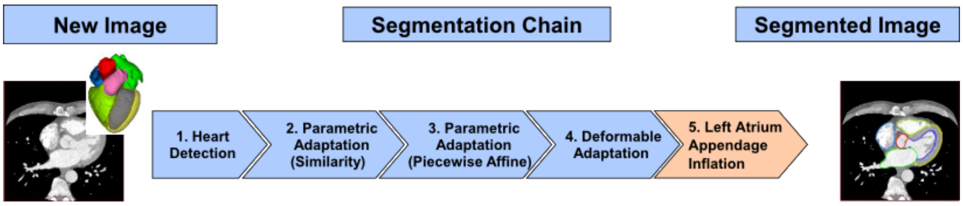

New Image
Segmentation Chain
Segmented Image
1. Heart Detection
2. Parametric Adaptation (Similarity)
3. Parametric Adaptation (Piecewise Affine)
4. Deformable Adaptation
5. Left Atrium Appendage Inflation

Figure 12: My work in the segmentation chain in the Philips method

- The model is adapted before the LAA begins to be segmented. The position of the base of the LAA is therefore known, and most of the surrounding structures (myocardium, vessels) are segmented.

4.2 Manual Segmentation

My first task was to segment manually the LAA for 17 patients. By doing so we can have ground truth which we can compare our automatic methods with.

For this I used an internal software from Philips called SegTools (screenshot figure 13). I annotated manually the voxels belonging to the LAA, then applied a smooth filter, and finally corrected by hand the remaining mistakes.

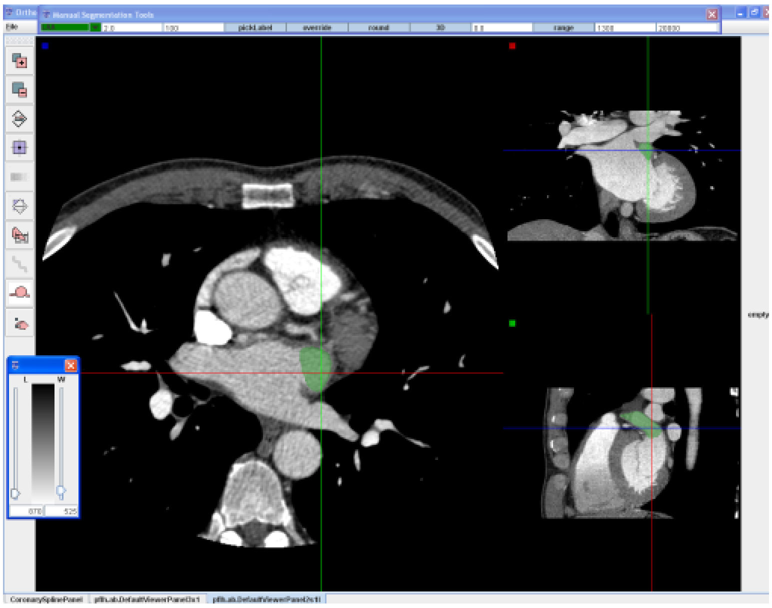

Ortho
Manual Segmentation Tools
File
LAA
pickLabel
override
round
3D
range
empty
L
W
CoronarySplinePanel
pfllLab.DefaultViewerPanel3x1
pfllLab.DefaultViewerPanel2x1

Figure 13: The SegTools software, with the LAA segmented in green

4.3 Mesh Modification

One important step in my work was to modify the model of the heart already created. I used for it the software MeshLab [2]. I made two changes on the 3D model:

- Closing of the Left Atrium in a "smooth way", trying to follow the natural curvature of the left atrium.

- Creation of another mesh and combination of the interfaces. This mesh will subsequently inflate into the LAA.

The first point was thus to define an interface between left atrium and LAA. Note that this interface doesn't exist in the heart, it has been made for practical reasons. The triangles are annotated "Left Atrium" on the inner side and "Left Atrial Appendage" on the outer side (see figure 16).

Then, I created a new mesh for the LAA. There was a few rules for creating this new mesh:

- Make a convex mesh, to inflate from the beginning in the good direction.

- Avoid sharp angles as much as possible.

- Make a progressive subsampling, with an higher density of triangles in the middle than on the borders. Indeed, the distribution of triangles has to be higher where the mesh will inflate the most to have a final regular triangle distribution.

The triangles in this mesh have to be quite small, in order to be at a reasonable size once the inflation is done. This part took me a significant amount of time, because it is a critical parameter on the quality of the inflation. The triangle here are annotated "Left Atrium" on the inner side, and nothing on the outer side.

The figures 14, 15, 16 and 17, show respectively the original mesh, the mesh without the original LAA, the mesh with the interface left atrium - LAA, and the mesh with the new LAA (before growing). Note that in the last picture, which represents the model I used, both of the interface and the new LAA are present, the interface is only hidden behind the LAA. The red color represents the removed (figure 14) or added triangles (figures 16 and 17).

Then I had to change some configuration files: with the old model, each triangle and each vertex is associated with meta-informations (features, labelling, internal energies, etc) in external files. When I change the mesh, the triangle correspondance is lost, therefore I had to modify these files. To do so I coded a few simple functions, with the help of Helko Lehmann, which detect the positions of the new triangles and vertices, the positions of the old triangles and vertices, find the correspondances, and modify the files according to these correspondances in an automated way.

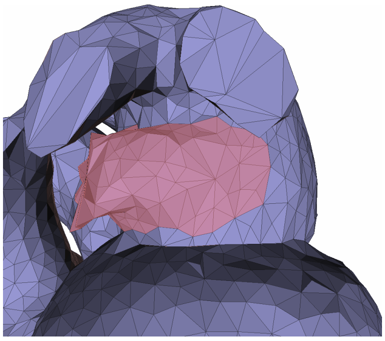


Figure 14: Original mesh

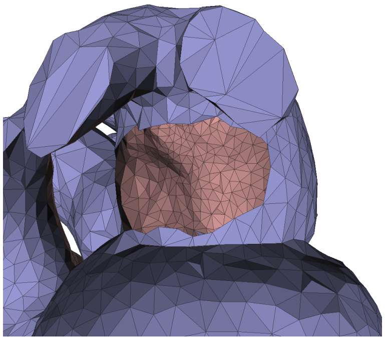


Figure 15: Mesh without the original LAA

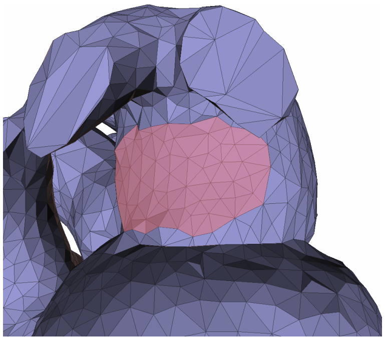


Figure 16: Mesh with the interface left atrium - LAA

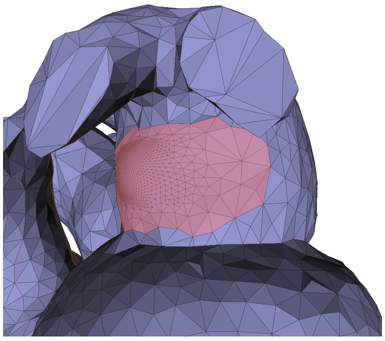


Figure 17: Mesh with the new LAA (before growing)

4.4 External and Internal Energies

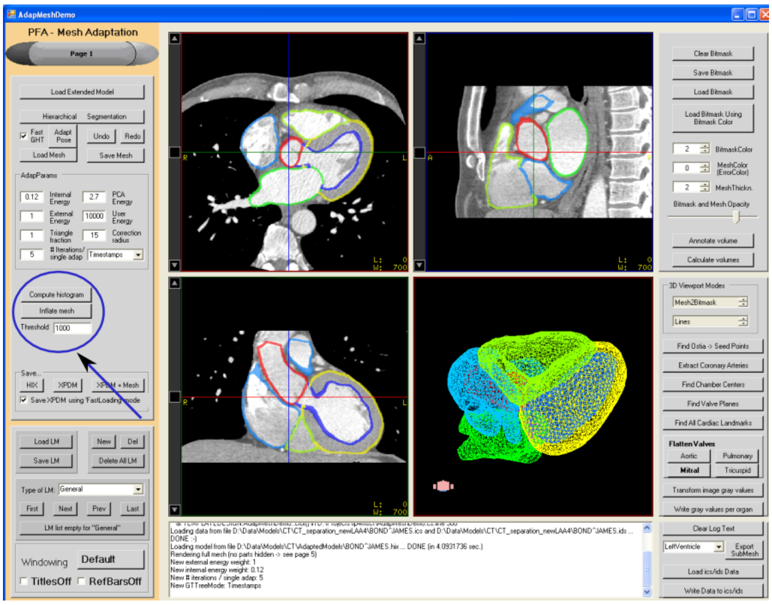
This task was mainly a coding task. Olivier added a few buttons in the C# interface for me, and from this I linked my own template and functions, in C++, in the main framework. The figure 18 shows a screenshot from this software.

As in the main method of the segmentation algorithm of Philips, the inflation method uses two types of energy: one external, which adapts the mesh to the image, and one internal, which constrains the mesh to avoid odd shapes and odd behaviours, all with a fixed topology. In this way, and contrary to less-constrained method like the geodesic active contours (see e.g. [16]), “leakages” of the mesh in other organs than the LAA are avoided. We have tried two different external energies and four different internal.

$$E = \alpha E_{\text{external}} + E_{\text{internal}} \tag{6}$$

where α a factor to balance the relative weight of the energies.

The figure 19 sums up the different energies. We could use one or any combination of the mentionned energies, with at least one external energy and one internal energy.



AdapParams
0.12 Internal Energy
2.7 PCA Energy
1 External Energy
10000 User Energy
1 Triangle fraction
15 Correction radius
5 # Iterations/ single adap
Timestamps

Threshold 1000

☑ Save XPDM using 'FastLoading' mode

Type of LM: General

2 BitmaskColor
0 MeshColor (ErrorColor)
2 MeshThickn.

3D Viewport Modes
Mesh2Bitmask
Lines

LeftVentricle

```
Loading data from file D:\Data\Models\CT\CT_separation_newLAA4\BOND^JAMES.ics and D:\Data\Models\CT\CT_separation_newLAA4\BOND^JAMES.ids ...
DONE :-)
Loading model from file D:\Data\Models\CT\AdaptedModels\BOND^JAMES.hix ... DONE (in 4.0931736 sec.)
Rendering full mesh (no parts hidden -> see page 5)
New external energy weight: 1
New internal energy weight: 0.12
New # iterations / single adap: 5
New GTTreeMode: Timestamps
```

Figure 18: AdapMeshDemo software. My buttons are in the left side (Compute Histogram, Inflate mesh), as shown with the arrow.

4.4.1 External Energies: Edge-based and Region-based

The external energy is separated in two parts: an edge-based energy, which is the same as in the current Philips method, and a region-based energy, which is detailed under.

Edge-Based Energy This energy is exactly the same as presented in the current method at Philips. It uses a special function, F_i, which detects specific boundaries. But I had to use default boundaries detector. This energy was thus attracted by any type of boundary, not necessarily the boundaries LAA/background. Indeed, the detector has to be trained before any adaptation, while keeping the topology of the mesh. The detector was not trained here, because it suppose a few *perfectly adapted* LAAs, and that was not our case. The specific detector being unavailable, and the default detector creating too many errors, we had to think about another energy.

Region-Based Energy This energy considers every vertex in the inflating LAA. The schemes 20 sums up the method.

The algorithm takes as initial point the center of gravity of each triangle. The gray value in this location[1] is evaluated by a tri-linear interpolation. This gray

[1] It is important to realize that there are two 3D data: the 3D mesh containing the coordinates of the triangles; and the 3D CT images containing voxels of different gray values. For this method we have to navigate between these two sets of data.

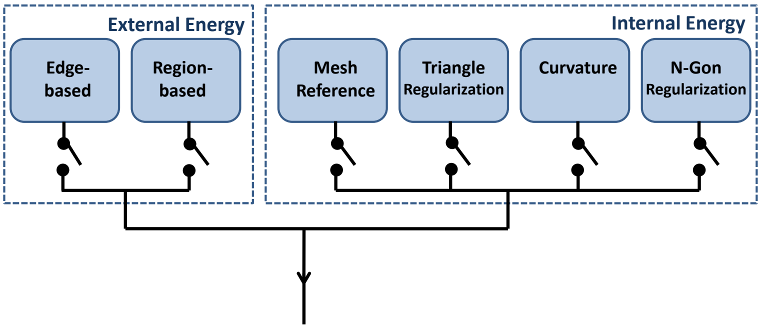


Figure 19: Summary of the different energies. Any combination is possible, as shown with the switches.

value is compared to a threshold which is computed as described in the section 4.4.2. This is the *(Step 0)*. If we are above threshold, the direction of growing will be outside (we suppose here that the triangles are pointing "outside" the mesh - that was the case with our mesh.) ; if under, the direction will be inside; if equal, the function returns a false.

Then, the algorithm tries to find a *target point*, the future position of the vertex. In the chosen direction (inside or outside mesh), following the normal of the triangle, it looks for 3 *candidate points*, at respectively 1, 2 and 3 mm from the initial vertex. Each time, two tests are done:

- Is the gray value still above (resp. under threshold) ?
- Is the voxel already annotated ?

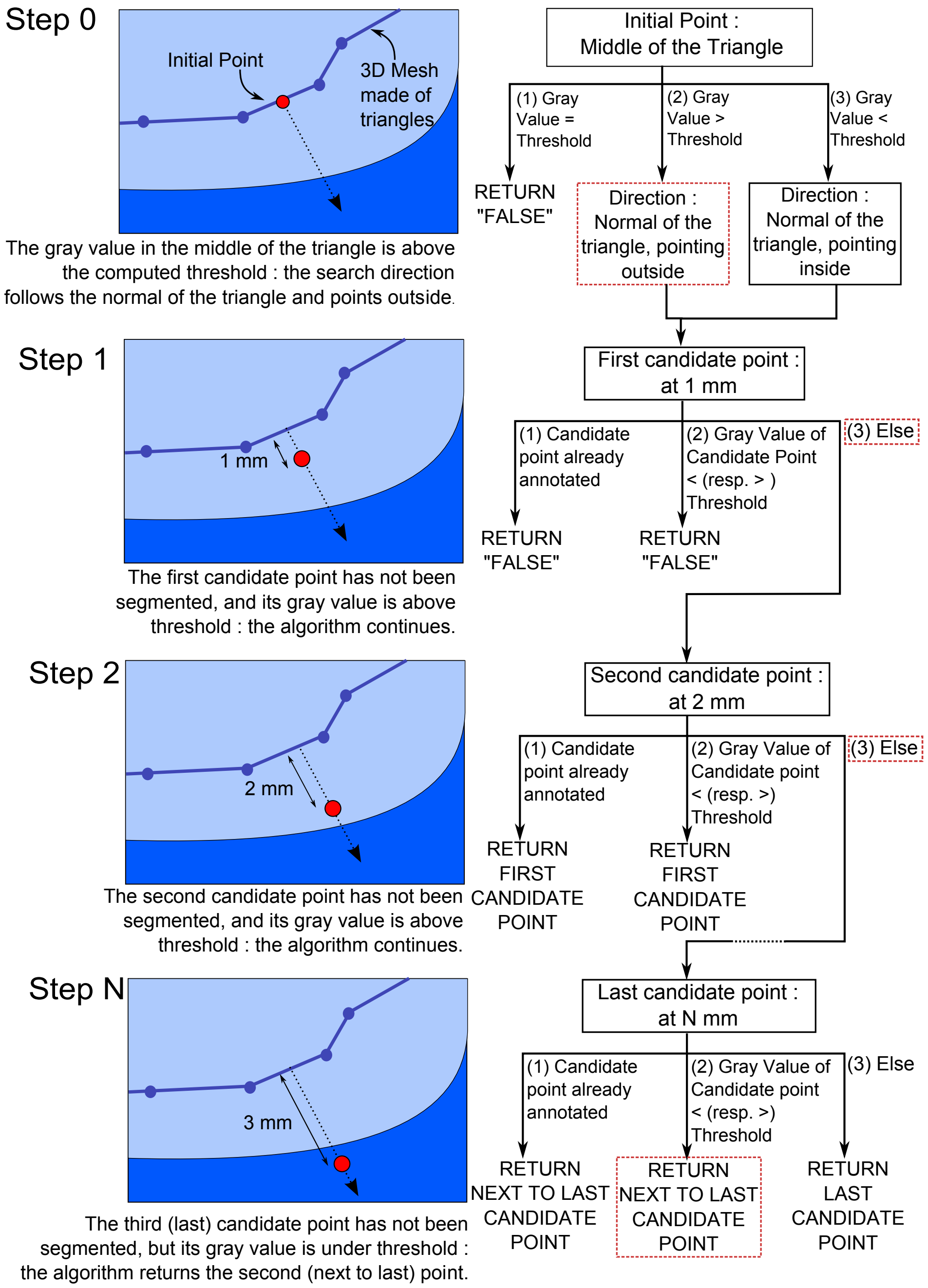
About the annotation, if the normal points *outside*, the mesh inflates only if the target point is not annotated, including annotated as LAA. If the normal points *inside*, the mesh shrinks only if the target point is not annotated, except if it is annotated as LAA. Indeed, we want to avoid as much as possible the overlapping of segmentation: as the growing factor is the gray value, the LAA could easily grow in another organ, without this test. The difference between growing and shrinking is due to the fact that we don't allow the LAA to overlap to itself.

For the first candidate point, at 1 mm, the algorithm sees thus if its gray value is on the "same side" as the initial point (that is to say, if the gray value of the initial point is above threshold, the gray value of the candidate point is too above threshold, and vice-versa), and if it is not segmented (*Step 1*). If both tests are positive, the algorithm looks at the second candidate point; if at least one answer is negative, the method stops and returns the value *"false"*.

For the second candidate point, at 2 mm, (*Step 2*) if both tests are positive, the algorithm looks at the *next candidate point*, 1 mm further; if at least one answer is negative, the method stops and returns the *previous candidate point* as a target point. This step can be looped until the last candidate point.

For the last candidate point, (*Step N*) if both tests are positive, the methods returns the *last candidate point* as a target point; if at least one answer is negative, the method stops and returns the *next to last candidate point* as a target point.

Note that in our algorithm, we have experimentally chosen 3 candidate points (so the Step 2 is done only 1 time), and 1 mm between candidate point. It has been revealed to be a good compromise between growing slowly, without beeing stuck by a darker voxels due to noise.



Step 0
Initial Point
3D Mesh made of triangles
Initial Point : Middle of the Triangle
(1) Gray Value = Threshold
(2) Gray Value > Threshold
(3) Gray Value < Threshold
RETURN "FALSE"
Direction : Normal of the triangle, pointing outside
Direction : Normal of the triangle, pointing inside
Step 1
1 mm
First candidate point : at 1 mm
(1) Candidate point already annotated
(2) Gray Value of Candidate Point < (resp. >) Threshold
(3) Else
RETURN "FALSE"
RETURN "FALSE"
Step 2
2 mm
Second candidate point : at 2 mm
(1) Candidate point already annotated
(2) Gray Value of Candidate point < (resp. >) Threshold
(3) Else
RETURN FIRST CANDIDATE POINT
RETURN FIRST CANDIDATE POINT
Step N
3 mm
Last candidate point : at N mm
(1) Candidate point already annotated
(2) Gray Value of Candidate point < (resp. >) Threshold
(3) Else
RETURN NEXT TO LAST CANDIDATE POINT
RETURN NEXT TO LAST CANDIDATE POINT
RETURN LAST CANDIDATE POINT

The gray value in the middle of the triangle is above the computed threshold : the search direction follows the normal of the triangle and points outside.

The first candidate point has not been segmented, and its gray value is above threshold : the algorithm continues.

The second candidate point has not been segmented, and its gray value is above threshold : the algorithm continues.

The third (last) candidate point has not been segmented, but its gray value is under threshold : the algorithm returns the second (next to last) point.

Figure 20: Schemes of the candidate search point method. In our algorithm we choose experimentally a distance of 1 mm between candidate points and a total of 3 candidate points. Note for the second test in the candidate point: if the gray value of the initial point is **below** threshold, the algorithm stops if the gray value of the candidate point is **above** threshold, and *vice-versa*.

4.4.2 Histogram Fitting

One critical parameter described in the region-based external energy is the segmentation between the voxels which belong to LAA and the voxels which don't. The most natural parameter is to consider the gray value of the voxels, and to find a gray-value threshold to separate them. The method presented here is fully automatic and is applied for each patient.

Around the LAA, we can mainly observe the left atrium (bright voxels of similar intensity as the LAA), Myocardium (darker voxels than LAA), blood and lungs (much darker voxels than LAA). Our approach was thus to find the best threshold between the gray values of left atrium voxels and the gray values of myocardium voxels.

As the model is already adapted, the algorithm can compute two histograms, one of Myocardium-voxels and one of Left Atrium-voxels. These two histograms are simply two arrays of 4096 bins (the total number of gray values possible). To compute these histogram I ran through the voxels of the image, and when one is annotated "Myocardium" (resp. "Left Atrium"), the corresponding gray value in the array of the Myocardium (resp. Left Atrium) is increased by one.

Two approaches have been implemented to find the best threshold between these two histograms.

Gaussian Fit Both histograms of myocardium-segmented voxels and the left-atrium-segmented voxels show in most of cases a bell function (see figure 21). Our first approach was thus to fit these distributions by Gaussian. Then, these Gaussians can be equalize to find their intersection, with A_1 and A_2 gaussian prefactors, σ_1 and σ_2 respective standard deviations and μ_1 and μ_2 respective x-values of the peaks: $\frac{A_1}{\sqrt{2\pi}\sigma_1} \exp \frac{-(x-\mu_1)^2}{2\sigma_1^2} = \frac{A_2}{\sqrt{2\pi}\sigma_2} \exp \frac{-(x-\mu_2)^2}{2\sigma_2^2}$

Simplifying this equation leads us to a second-degree equation (7) or a first degree equation, depending of the values of the standard deviation; we have thus either 0, 1, 2 or infinite solutions. In most cases we had naturally 2 solutions, and we had to choose the good result.

$$\Leftrightarrow (\frac{1}{\sigma_1^2} - \frac{1}{\sigma_2^2})x^2 - 2(\frac{\mu_1}{\sigma_1^2} - \frac{\mu_2}{\sigma_2^2})x + 2\ln(\frac{A_2\sigma_1}{A_1\sigma_2}) + (\frac{\mu_1^2}{\sigma_1^2} - \frac{\mu_2^2}{\sigma_2^2}) = 0 \qquad (7)$$

This method suffers of two problems: first, the choice of the good intersection between the two solutions, and more important, the supposition of gaussian shape. The histogram can be not perfectly gaussians, as shown on the figure 21. This led to another method called "Minimization of Classification Error", suggested by Jochen Peters from a paper by [20].

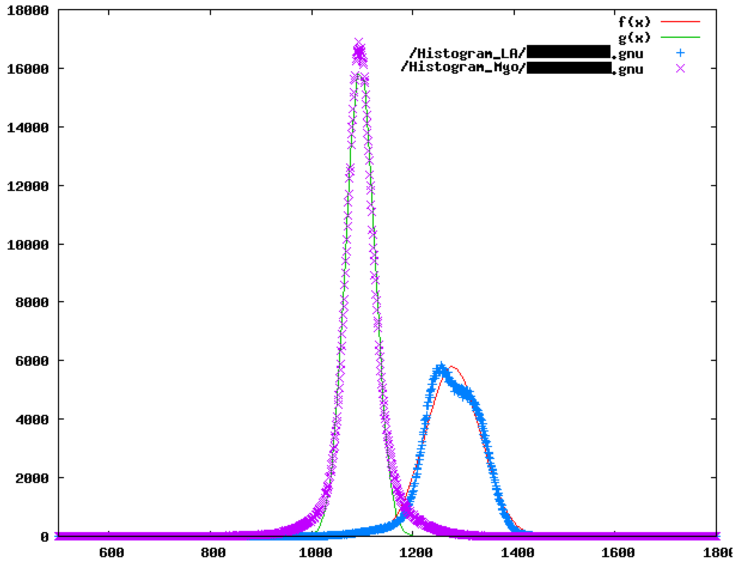


Figure 21: Histogram of Myocardium (violet points) and Left Atrium (blue points) with respective non-normalized gaussian fit.

Minimization of Classification Error In this method, summarized in the schemes 22 and 23, we consider the areas under the two histograms. We set a threshold at 0, and we increment it by 1. For each iteration, it checks for the area of the *first* histogram in its *right*, and for the area of the *second* histogram in its *left* - so the classification error for both histograms. At the beginning, the first area is naturally far bigger than the second. This algorithm stops when the second area becomes bigger than the first one, and return the found threshold. The classification error is thus minimized. Note that there wouldn't be any change if we switched the role of the two histograms.

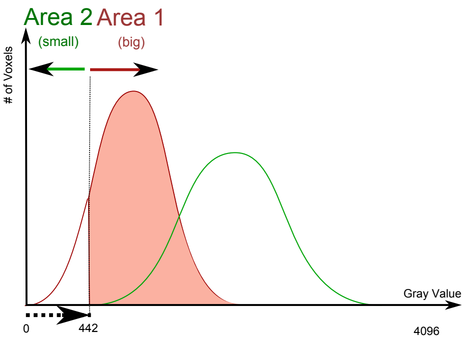


Figure 22: Beginning of the method "Minimization of Classification Error": the red area is bigger than the green one.

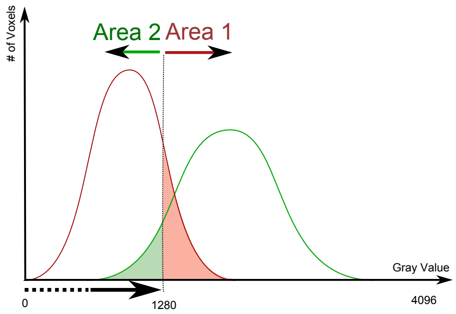


Figure 23: End of the method "Minimization of Classification Error": the red area is almost equal to the green one.

This method has revealed itself quick and efficient. Especially, we don't need to compute at each iteration the whole areas, we can just modify the previous ones by the number of voxels of the current gray value. The found thresholds were satisfying for all the tested patients.

4.4.3 Internal Energies: Triangle Regularization, Curvature, N-Gon Regularization, Mesh Reference

Triangle Regularization Energy This energy tries to approximate all the triangles into equilateral triangles.

The algorithm looks for all triangles. For each triangle, an equilateral triangle is created, with its vertices in (-$\frac{1}{2}$, 0, 0), ($\frac{1}{2}$, 0, 0) and (0, 0, $\frac{\sqrt{3}}{2}$). It is then rotated and isotropically scaled to minimize the distance between the corners of the initial triangle and the corners of the equilateral triangle (see scheme 24). The vertices of the initial triangle are thus subjected to forces which "pull" them into an equilateral triangle (strings on the schemes).

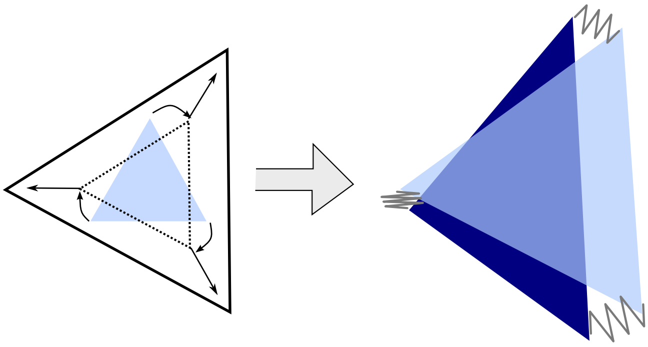


Figure 24: Scheme of the Triangle Regularization Energy: each triangle is approximated by a rotated and scaled equilateral triangle. Then some forces (strings on the schemes) are trying to deform the current triangle.

This energy can be written in this way:

$$E_{\text{triangle regularization}} = \sum_{i=1}^{V} \sum_{j=1}^{3} \left(v_{i,j} - v_{i,(j+1)mod.3} - \mathcal{T}^* \left[t_j - t_{(j+1)mod.3} \right] \right)^2 \quad (8)$$

where T^* is the optimized transformation, $v_{i,j}$ the j^{th} corners of the triangle i and the t_j the 3 corners of the equilateral triangle. The "mod.3" is to indicated that the 4^{th} corner of the triangle is identical to the 1^{st}.

Curvature Energy During the inflation, we often found some peaks, which can degenerate after (see scheme 25). Indeed, once a loop is created, the external energy tends to make it worse: as we suppose that the triangle points outwards, the external energy transform it in the "wrong way". We have thus thought about an energy which would remove peaks and smooth surface. We called this energy "curvature energy".

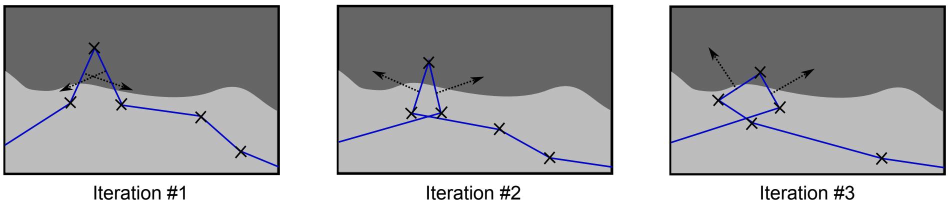


Figure 25: Schemes of a possible degeneration.

The principle, as shown in figure 26, is to looks for all the vertices. The algorithm takes one initial vertex and searches for its neighboors. Then, it computes a plane which goes through the neighboors (if there are more than 3 neighboors, the algorithm computes the best fitting plane with a least-square method). It searches then for the normal to this plane which is going through the initial vertex. The initial vertex will finally be moved in the intersection of the normal and the plane.

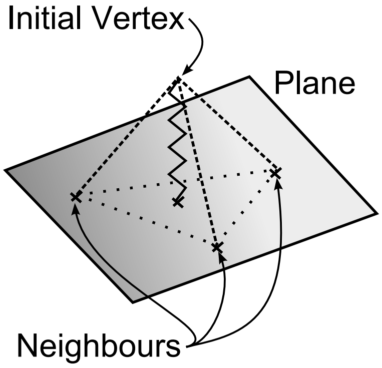


Figure 26: Schemes of the curvature energy algorithm.

This energy can be written this way:

$$E_{\text{curvature}} = \sum_{i=1}^{V} \sum_{j \in N(i)} \left[\mathcal{N}(N(i))^T \cdot (\boldsymbol{v}_j - \boldsymbol{v}_i)\right]^2 \tag{9}$$

where N(i) are the neighbors of the i^{th} vertex and $\mathcal{N}(N(i))$ is the vector normal to the plan passing through the neighbors vertices.

N-Gon Regularization Energy This energy was created because of an observation: most of the loops are appearing at the base of the mesh, in the interfaces between “low-density” triangles and “high-density” triangles. At the interface, the mesh is not regular (triangles with 12 neighbours), but have more neighbours on one side; if we try to make all triangles equilateral, heavy forces are present here and can make appear some loops (triangles on the wrong side) which degenerate afterwards.

Jochen Peters has thus suggested us an idea. The principle is very similar to the triangle regularization energy. Instead of considering triangles, it looks at the vertices. We call these vertices “N-Gon”, with N the number of neighbours of the reference vertex. For each N-Gon, we count the number of neighbours, and we create a regular N-Gon - that is to say, regularly distributed vertices at an equidistance from a reference vertex. This regular N-Gon is then rotated and scaled to be as close as possible from the initial N-Gon. The initial N-Gon is then replaced by the transformed N-Gon (see scheme 27).

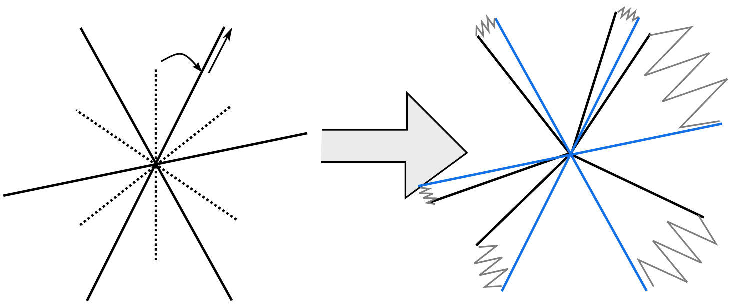


Figure 27: Scheme of the N-Gon Regularization Energy: each N-Gon is approximated by a rotated and scaled regular N-Gon. Then some forces (strings on the schemes) are trying to deform the current N-Gon.

This energy can be written in this way:

$$E_{\text{N-Gon regularization}} = \sum_{i=1}^{V} \sum_{j \in N(i)} \left((\boldsymbol{v}_i - \boldsymbol{v}_j) - (\mathcal{T}^*[\boldsymbol{g}_i - \boldsymbol{g}_j]) \right)^2 \qquad (10)$$

where T^* is the optimized transformation (linear), g_i the vertex in the center of the regular N-Gon and g_j its neighbors.

Mesh Reference This energy is the same as in the current Philips method (see equation 5): it has a reference mesh, and each deviation from this mesh is penalized. There one difference however: in the LAA-inflation algorithm, the reference mesh is regularly updated to the the current mesh. That is, at each iteration, the LAA grows, and the mesh created becomes the reference mesh. This energy has been shown experimentally to be the best internal energy that we could use.

4.5 Levels of Subsampling

The framework developed by Philips makes any change in topology impossible without major changes in the source code. However, a particular form of "subsampling" is possible. Subsampled meshes are made from a fine mesh whose topology is transformed to become coarser. The change in topology is done by removing some vertices; all the other vertices stay the same. In order to choose the vertices which will be removed, one criteria is applied: the global shape has to change as little as possible. The final number of faces can be chosen by the user. This subsampling can hardly be done automatically for each patient, and has thus to be pre-computed.

What we wanted to do was to inflate a rough mesh through the LAA, refine it a little bit, inflate again, etc, until it converged to something satisfying. Indeed, when the mesh is inflating, we often see that the triangles at the tip of the LAA are large, and their density is not high enough to inflate with the external energy. If we could increase the number of triangles here, the mesh could inflate farther.

This idea was however quickly rejected: after a few level of subsampling, most triangles, particularly the ones which shouldn't move, become too small, and some instabilities (loops) appear.

4.6 Loop Repair During Growing

The growing made appear some loops around the mesh (see screenshot 28). To deal with this problem, the intersecting faces are detected, with a method created by [1]. It searches then for its neighbours to the N-th order, and applies to them an internal energy. Then, the intersecting faces are again detected, and the growing is allowed only if this number is small enough.

In my code, I used neighbours to the 3d order, and the mesh was not growing if there were more than 10 intersecting faces after repair. To perform the mesh repair, only one internal energy is used to relax the mesh in the vicinity of a loop. The internal energy used is the mesh reference, the other energies were doing bad corrections, with the neighboors triangles at the last order around the repaired loop becoming quite large (see screeenshot 29). I tried too a Laplacian smooth, but the results were worse.

I remarked though that after a correction, a loop is not completely repaired: it appears again at the next iteration.

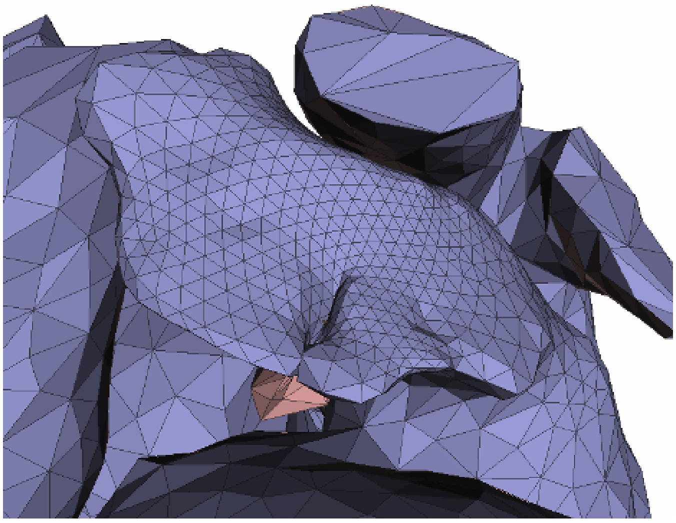


Figure 28: Screenshot of a loop in the mesh, in red. Some triangles are intersecting.

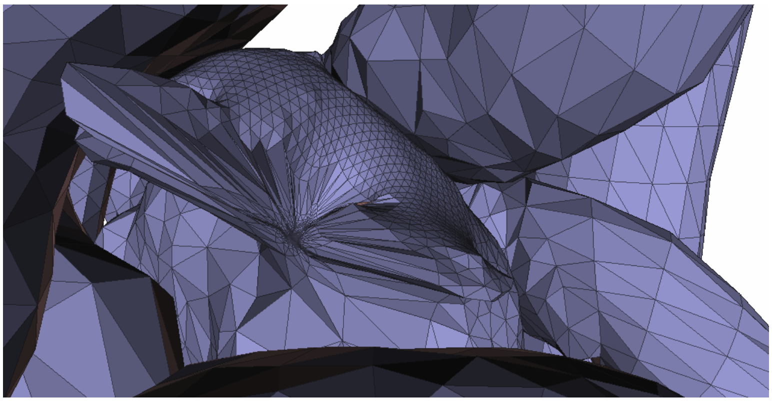
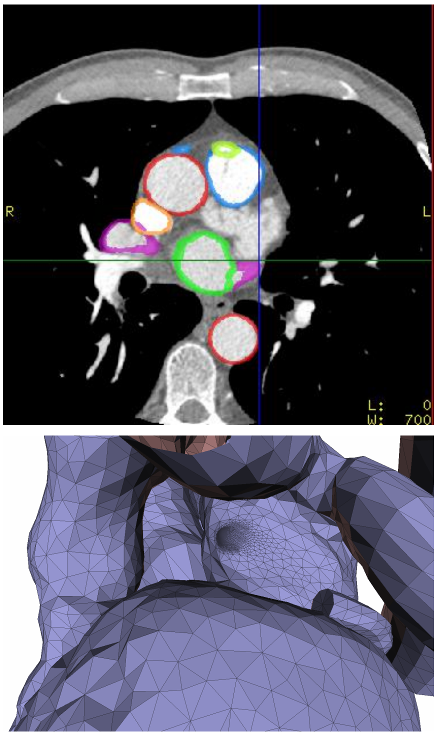


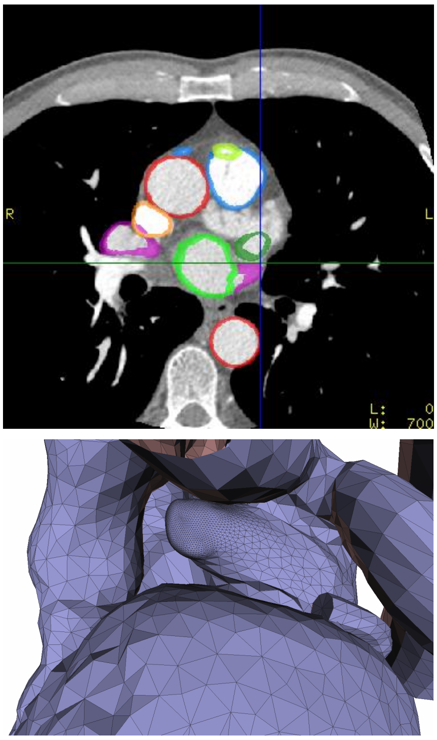
Figure 29: Screenshot of a bad repaired loop in the mesh. The triangle regularization with curvature energies were used.

5 Results

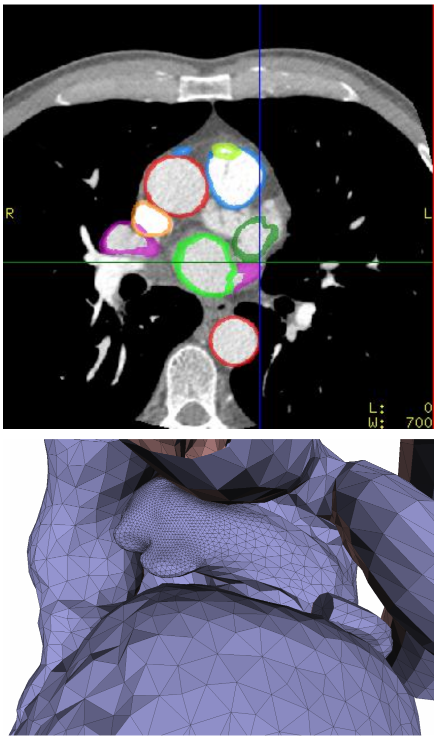
Here is an example of growing after 5 iterations between each picture. The internal energy used is the mesh reference only, and the external energy the region-growing only. During the growing, the factor α of the external energy was increasing every 5 iterations, from 0.2 to 5 (see equation 6)



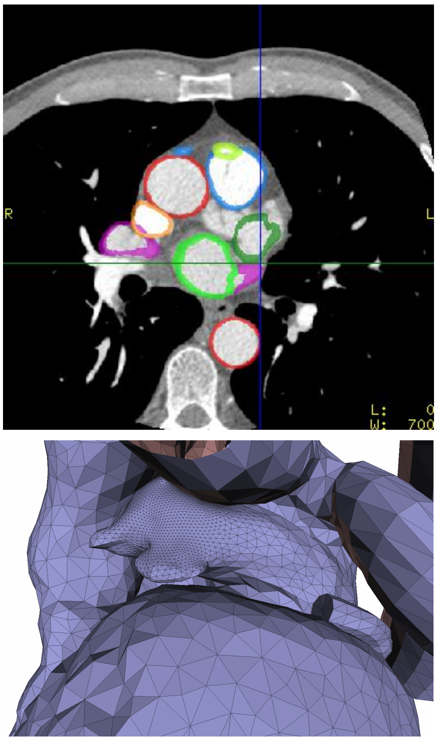
Before growing



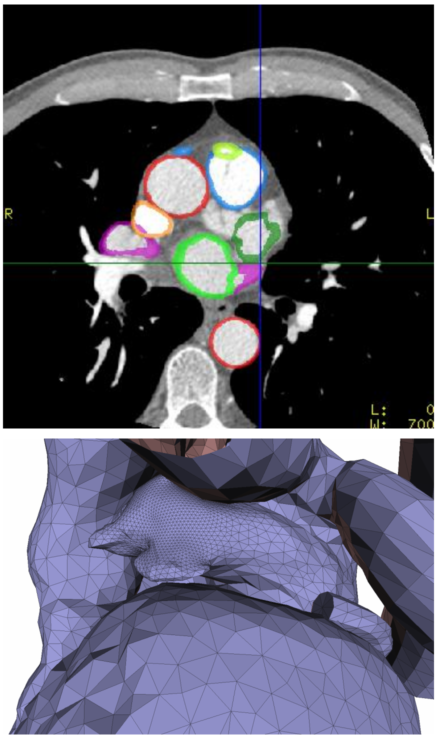
$\alpha = 0.2$



$\alpha = 1$



$\alpha = 2$



$\alpha = 5$

I made a lot of manual tests to find the best parameters to use: mesh, energies, values.

5.1 Detail of the Automatic Method Parameters

Mesh I have seen that the best mesh has an high density of triangles in the middle-left part of the left atrium appendage. I tried to avoid the interfaces between high-density and low-density triangles, because it is often the source of instabilities. The mesh is not very convex, just enough to go in the good direction. Indeed, I donn't see any visible improvment when the initial mesh is already a little inflated, while a flat mesh is more flexible between different patients. The figure 30 shows the current used mesh.

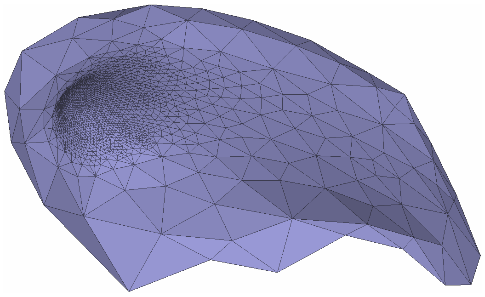


Figure 30: Close view of the mesh used for inflation.

Energies For the external energies, the edge-based energy is not helping so much, because it hasn't any trained features. Thus, any edge detected is taken, and that is not always the one we wanted. The region-growing energy seems to be reliable though.

For the internal energies, the mesh reference energy has been shown surprisingly efficient, for both of the growing and the loop correction. The growing is made step-by-step progressively, with the update of the reference mesh at each iteration, and it could explain this success.

The figure 31 illustrates which combination of energies has been experimentally proven to be the best.

Values There are 20 iterations in total. The mesh has often reached a steady state after 5 iterations, so the weight of the external energy is increased every 5 iterations. The external energy cannot indeed be very strong at the beginning, because the propagation becomes unstable, so it has to be become progressively stronger. The successive values of α (external energy weight) are thus 0.2, 1, 2 and 5.

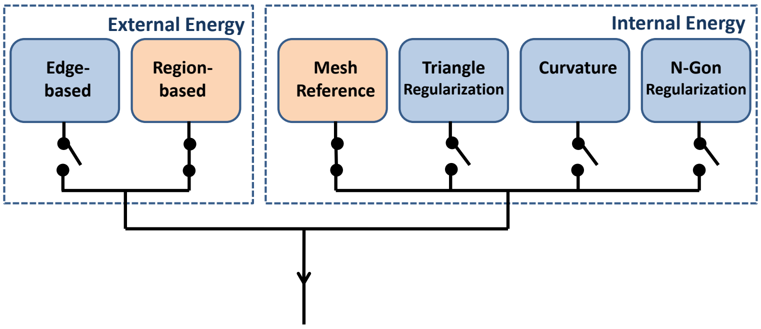
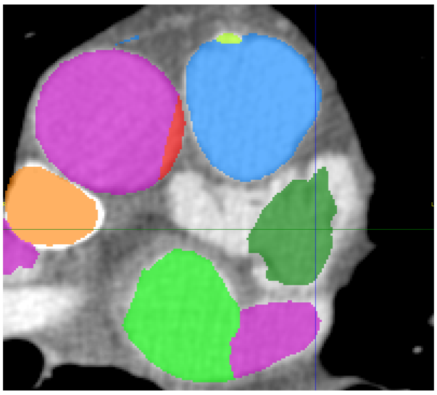


Figure 31: The used energies are shown in orange. This is the best combination that we found.

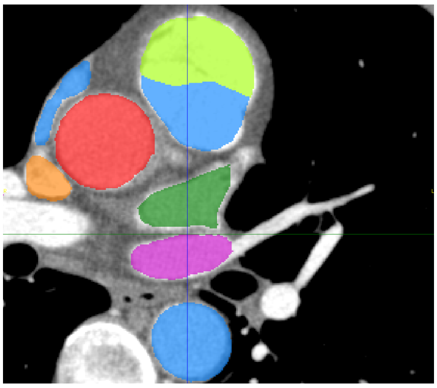
Computational Time The algorithm has been tested on a Intel Xeon at 2,4 Ghz, with 3,37 GB of RAM. The first 4 steps of the current method are taking about 20 seconds, and the inflation method between 20 and 40 seconds, depending on the patient.

5.2 Examples of Results

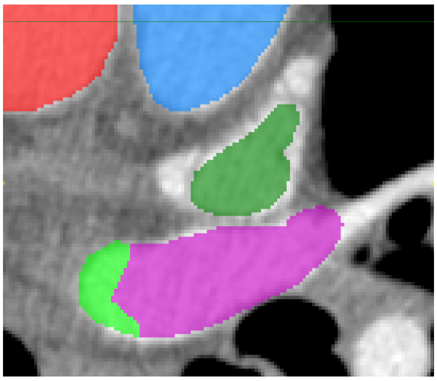
The images 5.2 have been taken from 4 different patients at "relevant position" of the LAA, close to the tip of the LAA. The patient number refers to their position in the graph 33.



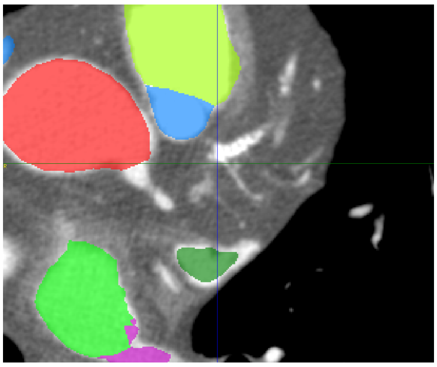
Patient (1)



Patient (7)



Patient (10)



Patient (5)

5.3 Numerical Evaluation

For evaluating the results, I simply look at each voxel enclosed by the mesh, and check its annotation: voxel belonging to the LAA (annotated during the manual segmentation) segmented by the algorithm, voxel belonging to the LAA not segmented by the algorithm, voxel not belonging to the LAA segmented by the algorithm, voxel not belonging to the LAA not segmented by the algorithm. The table below and the schemes 32 helps to visualize.

This technique of evaluating the adaptation doesn't take in account the quality of the adaptation: regular distribution of triangles, smooth shape... However, there shouldn't be any loop, as they are corrected by the algorithm when they appear.

	Annotated by Ground Truth	Not annotated by Ground Truth
Annotated by the algorithm	True Positive	False Positive
Not annotated by the algorithm	False Negative	True Negative

I then calculate some common values for each patient :

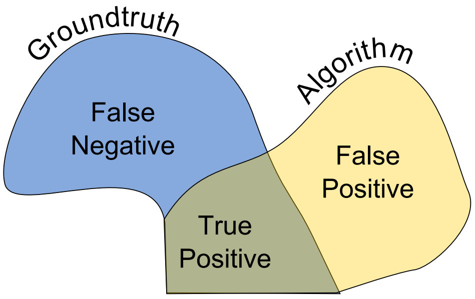


Figure 32: Reminder about true and false positive, and false negative.

- the specificity defined by: $\frac{TruePositive}{TruePositive+FalseNegative}$; it means: "If I take a voxels belonging to the LAA, which percentage of chance has it to be correctly annotated by the algorithm ?"
- the sensitivity defined by: $\frac{TrueNegative}{TrueNegative+FalsePositive}$; it means: "If I take a voxels not belonging to the LAA, which percentage of chance has it to be correctly annotated by the algorithm ?"
- A "quality" criteria equal to $\frac{TruePositive}{TruePositive+FalsePositive}$, which can be understood as "If I take a voxels segmented by the algorithm, what is the percentage of chance it really belongs to LAA ?"
- And the Dice coefficient, defined as $\frac{2*TruePositive}{FalsePositive+FalseNegative+2*TruePositive}$. It represents in one number the average quality of the segmentation.

As there are about 10^8 voxels in one 3D images, and the LAA is about 10^5 voxels, the sensitivity is always very close to 100%, and this common parameter is not very relevant. The Dice coefficient was always between the specificity and the quality - I didn't included it in the graph to improve its readibility. *NB: You can look at the appendix to see the raw results and these measurements.*

The main comment about these results is the difference between the specificity and the quality. **The mesh doesn't actually grow far enough, until the end of the LAA (low specificity), but almost all the voxels which are segmented really belong to the LAA (high quality).**

There is mainly one major fail, the patient 14 (very low specifity and quality), and three or four does not show satisfying enough results, like the patient 11 (which has rather high specificity). These fails are mainly due to some segmentation errors of the previous steps. In the case of the patient 14 - which had an anomaly in the left atrium vessels -, there is a pulmonary artery at the very base of the LAA (see image 34). The LAA doesn't grow at all in this case.

In the other cases, like for the patient 11, the left atrium and surrounding substructures are not exactly at the good place. For example, on the image 35, the left atrium, in bright green, shouldn't be visible at this height. The mesh has thus inflated in some wrong places.

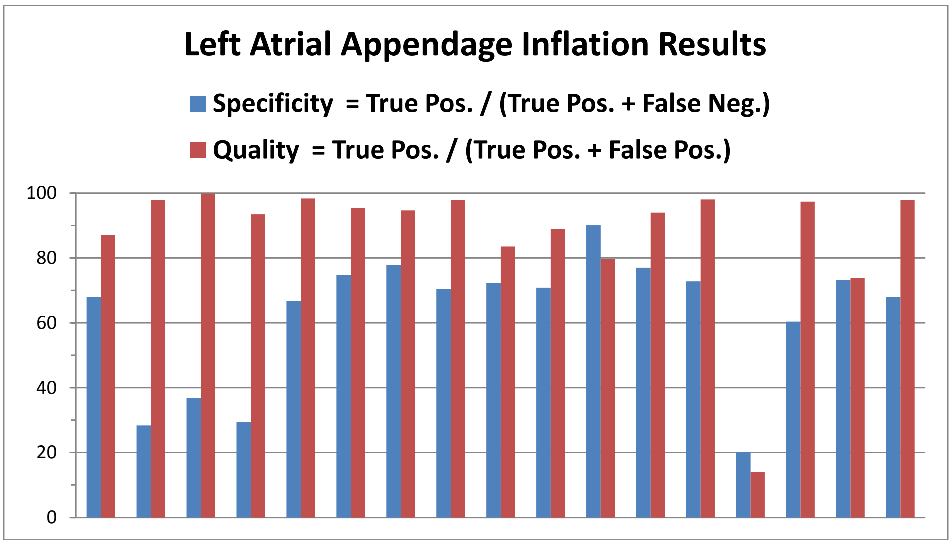


Figure 33: Numerical evaluation of the segmentation of the LAA by the algorithm.

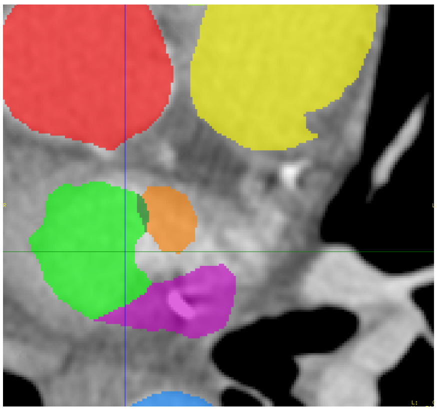


Figure 34: Patient 14: A pulmonary artery is present at the base of the LAA. The mesh doesn't grow at all.

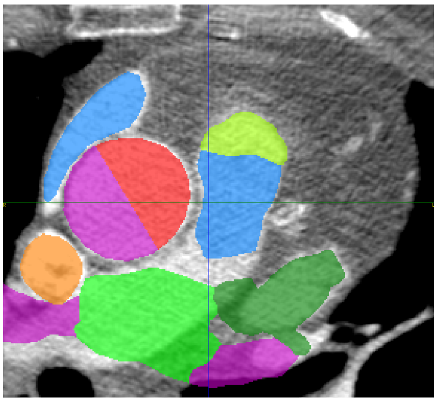


Figure 35: Patient 11: Previous modelization errors lead the mesh to grow in wrong places.

6 Conclusion and Future Work

The method developed during this master thesis is original. To my knowledge, this method has never been used before, even if it is based on methods which have proven their efficiency.

The current method is done in four steps, each one can repair small failures of the previous one: heart detection, global parametric adaptation, semi-global parametric adaptation, and deformable adaptation. They use for the 3 last steps an internal energy which penalizes any deformation of the mesh. For the last step, there is an external energy which detects the edges. The method is efficient, but can have small modelization errors.

The Left Atrial Appendage, highly deformable substructure of the heart, was segmented by using a method based on the Philips framework. This method inflates a flat mesh through the LAA, following an internal and an external energy. We tried four different energies: one called *Mesh Reference* is the same as in the current Philips addres; another one called *Triangle Regularization* approximates the triangles by equilateral triangle; the third one, the *Curvature*, removes the peaks and smooth the mesh; the last one is the *N-Gon Regularization*, and approximates N-Gon by regular N-Gon. The best internal energy found is the Mesh Reference Energy.

The external energy used is a region-growing energy based on gray values, with a threshold found by minimizing the classification error between two histograms.

The results are satisfying in the majority of cases, but they could be improved. Especially, the mesh doesn't inflate far enough into the LAA.

The algoritm could be improved on the loop correction: find a new smoothing method to repair them better ? Freeze the vertices to prevent them of creating new loops ?

We could think about finding a new internal energy which avoid the loops, and make a better distribution of triangles. I personally don't think that we could have a better initial mesh if the global method is not modified.

7 Acknowledgements

I would like to warmly thank O. Ecabert, for his excellent supervision, his great interest for my work, and above all, his exceptional human qualities. I also would like to thank the whole XIS team, especially J. Peters and R. Kneser, for their support of my work. This research was supported by Philips Forschungslaboratorien, which gave me a pleasant environment of work and make me discover the industry world. In addition, I wish to thank P. Cignoni and his team for creating the MeshLabs software, which is, despite a too small documentation, a powerful 3D software. O. Devillers and P. Guigue created an efficient algorithm which detects intersections of triangles. Finally, I wish to cite my teacher in the MVA, H. Delingette, who gave me the motivation to make my career in medical imaging, and who helped me to find an master thesis with all the criteria I requested (and it was not so easy...) !

A Annex: Raw Results per patient

A.1 Voxels Annotation

Patient	True Positive	False Positive	False Negative	True Negative
1	59688	8816	28159	59934313
2	44889	983	113440	87134640
3	37796	37	64829	102657786
4	22969	1603	54787	70961665
5	71819	1221	35823	65164993
6	47268	2254	15914	70451300
7	154726	8729	44093	130340164
8	78106	1714	32765	82200631
9	38685	7595	14760	67309968
10	65288	8125	26822	96368757
11	131434	33611	14482	87114425
12	62330	3985	18568	69121133
13	233689	4555	87167	76220637
14	15846	96847	62641	105468698
15	22220	601	14559	131296764
16	12378	4383	4538	50572493
17	64372	1399	30429	122587192

A.2 Computed Measurments

Patient	Specificity	Sensitivity	Quality Criteria	Dice Coefficient
1	67.9454	99.9853	87.1307	76.3513
2	28.3517	99.9989	97.8571	43.9655
3	36.8292	100	99.9022	53.8182
4	29.5398	99.9977	93.4763	44.8929
5	66.7202	99.9981	98.3283	79.4977
6	74.8124	99.9968	95.4485	83.8799
7	77.8225	99.9933	94.6597	85.4193
8	70.4476	99.9979	97.8527	81.9189
9	72.3828	99.9887	83.589	77.5834
10	70.8805	99.9916	88.9325	78.8869
11	90.0751	99.9614	79.6353	84.5341
12	77.0476	99.9942	93.9908	84.68
13	72.833	99.994	98.0881	83.5947
14	20.1893	99.9083	14.0612	16.577
15	60.4149	99.9995	97.3665	74.5638
16	73.1733	99.9913	73.85	73.5101
17	67.9022	99.9989	97.8729	80.1784

References

[1]

[2] Paolo Cignoni, Massimiliano Corsini, and Guido Ranzuglia. MeshLab: an Open-Source 3D Mesh Processing System. *ERCIM News*, (73):45–46, April 2008.

[3] Claudia Stöllberger, Birke Schneider, and Josef Finsterer. ”Elimination of the Left Atrial Appendage To Prevent Stroke or Embolism?”. *Chest 124*, 6:2356–2362, dec 2003.

[4] A. Hyvärinen and E. Oja. ”Independent Component Analysis: Algorithms and Applications”. *Neural Networks*, 13 (4-5):411–430, 2000.

[5] Jochen Peters, Olivier Ecabert, and Jürgen Weese. ”Feature learning for model-based image segmentation”. *PFL-A Technical Note 2005-00582, Philips Research Laboratories, Aachen*, July 2005.

[6] John P. Veinot, Phillip J. Harrity, Federico Gentile, Bijoy K. Khandheria, Kent R. Bailey, Jeffrey T. Eickholt, James B. Seward, A. Jamil Tajik, and William D. Edwards. ”Anatomy of the normal left atrial appendage: a quantitative study of age-related changes in 500 autopsy hearts: implications for echocardiographic examination”. *Circulation 96*, 9:3112–3115, 1997.

[7] Joseph L. Blackshear and John A. Odell. ”Appendage Obliteration to Reduce Stroke in Cardiac Surgical Patients With Atrial Fibrillation”. *Ann. Thorac. Surg.*, 61:755 – 759, Feb 1996.

[8] Jürgen Weese, V. Pekar, M.R. Kaus, C. Lorenz, and P. Rösch. ”3D medicale image segmentation with shape constrained deformable models”. *PFL-H Report 875/00, Philips Research Laboratories, Hamburg*, dec 2000.

[9] M.R. Kaus and V. Pekar. ”Automated organ delineation for radiation therapy planning”. *PFL-H Technical Note 42/03, Philips Research Laboratories, Hamburg*, jan 2004.

[10] J. Lötjönen, S. Kivistö, J. Koikkalainen, D. Smutek, and K. Laurema. ”Statistical shape model of atria and ventricles and epicardium from short- and long-axis MR images”. *Medical Image Analysis*, vol 8:371–286, 2004.

[11] T. McInerney and Demetri Terzopoulos. ”Deformable models in medical image analysis: a survey”. *Medical Image Analysis*, 1(2):91–108, 1996.

[12] Michael Kass, Andrew P. Witkin, and Demetri Terzopoulos. ”Snakes: Active contour models”. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[13] Olivier Ecabert, Jochen Peters, and Jürgen Weese. ”Modeling shape variability for full heart segmentation in cardiac computed-tomography images”. *Proc. of SPIE Medical Imaging*, 2006.

[14] Olivier Ecabert, Jochen Peters, H. Schramm, C. Lorenz, J. von Berg, M. Walker, M. Vembar, M. Olszewski, K. Subramanyan, G. Lavi, and Jürgen Weese. ”Automatic Model-Based Segmentation of the Heart in CT Images”. *IEEE Transactions on Medical Imaging*, 27, sept 2008.

[15] H. C. van Assen, M. G. Danilouchkine, M. S. Dirksen, J. H. C. Reiber, and B. P. F. Lelieveldt. ”A 3-D Active Shape Model Driven by Fuzzy Inference: Application to Cardiac CT and MR”. *IEEE Transactions on Information Technology in Biomedicine*, 12 and num. 5, sept 2008.

[16] Vicent Caselles, Ron Kimmel, and Guillermo Sapiro. Geodesic Active Contours. *International Journal of Computer Vision*, 22:61–79, 1995.

[17] J von Berg, M.R. Kaus, V. Pekar, and A. Franz. ”Modelling gray value aprearance for object surface detection”. *PFL-H Report 1631/2003, Philips Research Laboratories, Hamburg*, feb 2003.

[18] A. A. Young and A. F. Frangi. ”Computational cardiac atlases: from patient to population and back”. *Experimental Physiology*, 94.5:578–596, april 2009.

[19] Y. Zheng, A. Barbu, B. Georgescu, M. Scheuering, and D. Comaniciu. ”Four-Chamber Heart Modeling and Automatic Segmentation for 3D Cardiac CT Volumes Using Marginal Space Learning and Steerable Features”. *IEEE Transactions on Medical Imaging*, 2008.

[20] Y. Zheng, B. Georgescu, F. Vega Higuera, and D. Comaniciu. ”Left Ventricle Endocardium Segmentation for Cardiac CT Columes Using an Optimal Smooth Surface”. *Medical Imaging 2009 : Image Processing - Proc. of SPIE*, 7259, 72593V, 2009.