4D Cardiac MRI Segmentation and Surface Reconstruction

Gerard Pons Moll
Department of Electrical and Computer Engineering
Northeastern University

A thesis submitted for the degree of
Telecommunications Engineering Masters Thesis

2008 July
1. Reviewer: Name

2. Reviewer:

Day of the defense:

Signature from head of the committee:
Abstract

Atrial fibrillation (AF) is a growing problem in modern societies with an enormous impact in both short term quality of life and long term survival. A recently developed promising approach to cure AF uses radiofrequency (RF) ablation to carry out "pulmonary vein antrum isolation" (PVAI) to the heart. However, the lack of proper 3D visualization during surgery training, planning, and guidance makes surgery a very difficult task to the surgeons and therefore the risk for the patient increases.

The purpose of this work is to develop methods for automatically segmenting and tracking the heart in 4-D cardiac MRI datasets. The heart surface will be reconstructed to serve as a virtual computer model for the 3D surgery training, planning and guidance. In this work we also outline some of the methods proposed to solve the problem of cardiac segmentation. Our method is based on an active contour model and a geometric post-processing of the segmentation.
La fibril·lacó auricular és un tipus de arítmia cardíaca i es un problema creixent en la societat moderna amb un impacte enorme en la qualitat de vida a curt termini així com en les probabilitats de supervivència a llarg termini. Una tècnica cirúrgica prometedor desenvolupada recentment consisteix en dur a terme l’ablació per radiofreqüència a una regió específica del cor. Amb la tecnologia i software actual, el cirurgians han de realitzar l’ablació per radiofreqüència amb un catèter i la única visualització de la que disposen són imatges en 2D provinents de una càmera endoscòpia instal·lada a l’extrem del catèter. Normalment el camp de visió es molt reduït i la navegació es en 2D, la qual cosa fa que la operació sigui extremament difícil i per tant el risc per al pacient incrementa.

La falta de visualització en 3D en les etapes de entrenament, planificació i guiat de la cirurgia cardíaca ha motivat la realització d’aquest projecte presentat en aquest treball. L’objectiu principal d’aquest projecte es extender l’entorn de treball del cirurgià cardíac creant un entorn virtual en el qual es pugui navegar amb el catèter visualitzant en pantalla un model tridimensional del cor específic pel pacient. Per a que això sigui possible el problema es pot dividir en 2 etapes diferenciades.

1. Creació de un model pre-operatiu 3D+temps del cor específic per al pacient.

2. Alinear el sistema de coordenades del model amb les imatges disponibles a la sala d’operacions.

Si es poden solucionar aquests dos problemes el model virtual del cor creat abans de la operació estarà perfectament alineat amb la posició del cor real del pacient a la

---

Durant la realització del Projecte Final de Carrera l’autor estava en el departament de Biomedical Signal Processing a Northeastern University, Boston, USA
sala d’operacions.
El treball presentat en aquesta tesi consisteix en desenvolupar un mètode per crear automàticament el model del cor a partir de les imatges de una sessió de ressonància magnètica (MRI) del pacient pre-operativa. Les ressonàncies magnètiques amb les que treballem consisteixen en volums formats per diversos talls transversals del cor. Per cada fase cardíaca s’obté un d’aquests volums formant per tant un conjunt de dades en 4D del cor.
El mètode esta basat en un contorn actiu per a la localització, segmentació i seguiment del cor durant diferents fases del cicle cardíac. Un cop obtinguts els contorns que defineixen la paret exterior del cor per totes les imatges transversals es proposa un processat geomètric per donar consistència temporal i espaial a les segmentacions obtingudes. Un cop processada la data es procedirà a la reconstrucció de la superfície cardíaca a partir de les segmentacions planars processades. Finalment, es visualitzarà el moviment de les superfícies obtingudes de la paret exterior del cor i del ventricle esquerre.

**Breu descripció del mètode:**

S’inicialitzarà el contorn actiu introduint manualment 5-6 punts per cada un dels talls transversals per a la primera fase del cicle cardíac. Aquests punts seran interpolats utilitzant splines i es mostrejarà el contorn amb 5 graus de separació respecte el centroïde de la corba formant així un total de 72 nodes per contorn. Des d’aquest punt endavant el mètode haurà de pertorbar els nodes en la direcció radial fins que s’enganxin a les paret del cor en la imatge.

Per tant es buscarà en les línies radials del contorn, característiques locals de les imatges per trobar la paret exterior del cor. Quan els 72 nodes s’hagin enganxat a la paret del cor els projectarem a l’espai de Fourier i ens quedarem només amb els primers 5 harmònics per tal d’obtenir una representació suau de la paret del cor. Quan s’hagi trobat la frontera del cor per un determinat instant temporal es propagarà el contorn actual i es farà servir com a estimació per la següent fase cardíaca i repetirà el procés anterior. Un cop s’hagin obtingut tots els contorns per un determinat tall transversal s’aprendrà el moviment dinàmic del cor. El moviment dinàmic
estarà representat per una matriu que utilitzarem com a informació a priori per millorar les prediccions dels subsegüents talls transversals. Aquesta matriu de moviment s’actualitzarà per cada tall transversal.

Un cop s’hagin trobat tots el contorns, procedirem a fer un processat geomètric dels nodes que defineixen les superfícies temporals. Agruparem els 72 nodes de cada contorn en una matriu 3D. Cada columna descriu el contorn per a un determinat tall transversal i un determinat instant temporal. Es a dir, quan avancem en la matriu horitzontalment anem trobant els diferents contorns del volum del cor per a un mateix instant temporal i quan avancem en la direcció de profunditat trobem els contorns per les diferents fases cardíacas o instants temporals. Els contorns s’han trobat d’una manera mes o menys independent i per tant imposarem consistència temporal i espacial als nodes. D’aquesta manera restringirem el problema imposant a les superfícies que tinguin transicions suaus en cadascuna de les 3 dimensions espacials més la temporal. Per suavitzar els nodes en totes les direccions i eliminar el soroll creat per l’algorisme utilitzarem ”m-tensor smoothing splines”. Bàsicament, consisteix en encaixar una funció polinòmica a trossos a cadascuna de les dimensions que creuen un node; per tal i com havíem parametritzat les superfícies i la manera com havíem definit la matriu tridimensional les dimensions són:

- variació angular dins el contorn on es troba el node
- variació entre el node i els seus veïns en els talls superior i inferior
- variació entre un node i els seus veïns per l’instant temporal anterior i posterior

Un cop tinguem una funció polinòmica per cada una de les dimensions que travessen un node, el node serà substituït per el producte ponderat de les tres funcions que el travessen. Es pot pensar aquest problema com una reixa tridimensional, els eixos d’aquesta reixa es deformen d’una manera suau per apropar-se als nodes de la matriu 3D, d’aquesta manera cada node es substitueix per una ponderació de cadascun dels tres eixos que el travessen.

Un cop obtinguts els nodes suavitzats reconstruirem la superfície, la manera de connectar els nodes és implicita en la matriu tridimensional ja que els veïns dels nodes en la estructura de la superfície son també els veïns en la matriu. D’aquesta manera podrem visualitzar el moviment de les superfícies tridimensionals. Les superfícies obtingudes
ajudaran a diagnosticar anomalies en la contracció càrdiaca i podran servir, un cop es registrin amb les imatges operatives, com a model virtual per la visualització en 3D durant les etapes de la cirurgia cardíaca.
To Laura
Acknowledgements

I would like to acknowledge all the people who have helped me realize this thesis. Specially, I want to thank my supervisors, Dr. Dana H. Brooks and Dr. Gilead Tadmor. They have been really helpful and have given plenty of good ideas during many interesting discussions. I am specially grateful because they always found a moment to help me and I have enjoyed my research with them.

This thesis was conducted in collaboration with the University of Utah. I would like to thank Dr. Rob MacLeod for providing us the datasets and for the helpful advise.

I would also like to thank all the people in the Biomedical Image Processing group at Northeastern University for all the help and interesting group meetings.

I want to express my deep gratitude to my girlfriend Laura Leal who has been always very supportive. Without her I wouldn’t enjoy research as much as I do. Finally, I would like to thank my friends, and specially my parents and sister who always did their best in supporting me.
## CONTENTS

5  **Epicardium segmentation**

5.1 Understanding the data .......................... 29
5.2 Processing the data .............................. 31
  5.2.1 Preprocessing .................................. 31
  5.2.2 Initial segmentation .......................... 32
  5.2.3 Correct the boundary ......................... 35
  5.2.4 Curve parameterization and FFT ............. 38
  5.2.5 Propagate the contours ....................... 42
  5.2.6 Time smoothing ................................ 42
  5.2.7 Velocity estimation ........................... 45
    5.2.7.1 Velocity computation ..................... 46
  5.2.8 The code ..................................... 48

6  **Heart Surface Reconstruction** ............... 49

6.1 M-Tensor Product Splines ........................ 49
6.2 Smoothing splines ............................... 51

7  **Results** ....................................... 57

8  **Discussion and Future Work** ................ 67

8.1 Conclusions ..................................... 67
8.2 Reducing the complexity ......................... 68
8.3 Initialization of the algorithm .................. 68
8.4 Shape priors .................................... 69
8.5 Include heart rotation and axial shortening to the model .......................... 69
8.6 Respiratory motion ................................ 69
8.7 2D-3D geometric based registration ............ 70

References ......................................... 71
List of Figures

1.1 Temporal registration .......................................... 6
3.1 MRI ECG gated ................................................ 18
3.2 Dataset1 ...................................................... 19
3.3 Dataset2 ...................................................... 19
4.1 Outlier with high curvature ...................................... 23
4.2 High curvature points are erased ............................... 23
4.3 Orientation of the search lines ................................ 24
4.4 Particle travelling ............................................... 26
4.5 Wireframe model of the binary heart ......................... 27
5.1 Canine heart surface ........................................... 30
5.2 RV fills up ..................................................... 30
5.3 Canine heart surface ........................................... 31
5.4 Fat .......................................................... 31
5.5 Original image ................................................ 32
5.6 Output of opening ............................................. 32
5.7 Binary mask .................................................... 32
5.8 Final image .................................................... 32
5.9 Steps to clean the images ..................................... 32
5.10 Curved cylinder ............................................... 33
5.11 centroid align ................................................ 34
5.12 Sampling at 5 degrees separation ............................. 35
5.13 searchlines ................................................... 36
5.14 Figure shows how the boundary point is selected ........ 37
### LIST OF FIGURES

5.15 Image gradient ............................. 39
5.16 Shows the $r(\theta)$. The left plot shows the initial points and the plot in the right shows the guesses after keeping the first 5 harmonics ................. 40
5.17 Shows the $r(\theta)$. This plot shows how FFT is a good approximation and is not affected by the outliers ................................. 41
5.18 Shows the $r(\theta)$. The red points are the initial guesses and the blue points are after keeping the first FFT coefficients. .......................... 42
5.19 The blue curve is the contour in $t - 1$ and the red one is the corrected contour in $t$ ............................................................. 43
5.20 Shows the node trajectories and the polynomial smoothing .............. 44
5.21 Before polynomial fitting .................................................... 45
5.22 After Polynomial fitting ...................................................... 45
5.23 Surface of the velocity matrix .............................................. 47
6.1 We need to smooth the radius of the points in the 3 dimensions .... 50
6.2 3D matrix structure .............................................................. 50
6.3 Radius 3D matrix cross-section for $t = t_i$ before smoothing splines ... 55
6.4 Radius 3D matrix cross-section for $t = t_i$ after smoothing splines ...... 55
6.5 Radius 3D matrix crosssection for $\theta = \theta_i$ before smoothing splines .... 55
6.6 Radius 3D matrix crosssection for $\theta = \theta_i$ after smoothing .......... 55
7.1 Canine heart surface ............................................................. 58
7.2 Canine heart surface ............................................................. 59
7.3 Epicardium contours ............................................................ 60
7.4 LV contours and surfaces ..................................................... 61
7.5 LV surfaces ................................................................. 61
7.6 Epicardium and LV surfaces and contours ................................. 62
7.7 Epicardium wireframe in red and LV surface in blue ................. 63
7.8 View1: Epicardium and LV surfaces ........................................ 63
7.9 View2: Epicardium and LV surfaces ........................................ 63
7.10 View3: Epicardium and LV surfaces ....................................... 63
7.11 We can perfectly see the LV in this vertical cross-section ............. 64
7.12 LV mesh along with the data ................................................... 64
7.13 LV after adding faces to the mesh ........................................... 64
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.14</td>
<td>Mesh of both the epicardium in red and the LV in blue</td>
<td>64</td>
</tr>
<tr>
<td>7.15</td>
<td>Long axis cross-section</td>
<td>65</td>
</tr>
<tr>
<td>7.16</td>
<td>Long axis cross-section along with the epicardium wireframe</td>
<td>65</td>
</tr>
<tr>
<td>7.17</td>
<td>3D heart surface overlapped to the data</td>
<td>65</td>
</tr>
<tr>
<td>7.18</td>
<td>3D heart surface from a different view</td>
<td>65</td>
</tr>
<tr>
<td>7.19</td>
<td>Long axis cross-section with LV surface in blue</td>
<td>66</td>
</tr>
<tr>
<td>7.20</td>
<td>Epicardium and LV surfaces, we can see how they fit the data</td>
<td>66</td>
</tr>
<tr>
<td>7.21</td>
<td>Epicardium and LV wireframe models intersecting a short axis image</td>
<td>66</td>
</tr>
<tr>
<td>7.22</td>
<td>Epicardium wireframe and LV surface, we can clearly see how the aorta comes out the LV surface</td>
<td>66</td>
</tr>
</tbody>
</table>
List of Tables

3.1 dataset table .................................................. 17
Chapter 1

Introduction

1.1 Problem statement and Challenges

Atrial fibrillation (AF) is a growing problem in modern societies with an enormous impact in both short term quality of life and long term survival. Standard AF therapy, which includes antiarythmic drugs has been unsatisfactory with success on only 40-60 % of those treated. A recently developed promising approach to cure AF uses radiofrequency (RF) ablation to carry out "pulmonary vein antrum isolation" (PVAI). Unfortunately, because of significant clinical and technical challenges this approach is used in less than 0.1% of AF patients, often with suboptimal results. The hypothesis of this research is that current and future magnetic resonance imaging (MRI) will allow PVAI to become the standard treatment for AF. Conventional surgery is performed via a median sternotomy on the arrested heart, requiring the use of cardiopulmonary bypass (CPB), aortic cross clamping, and myocardial preservation. The latter are the main causes of side effects that lead to increased patient recovery times and costs. Unwanted side-effects, such as large incisions, CPB, or cardiac arrest can be reduced by performing cardiac surgery using an endoscope-aided, port-access approach described previously. In practice however, operating on the beating heart without direct vision is extremely challenging. The first problem is the requirement to perform complex maneuvers, normally carried out in the open chest, inside the closed thoracic cavity. Several systems have been effective for such tasks when used by a trained practitioner. The second and more prominent problem is the lack of proper three-dimensional (3D) visualization during the training, planning, and guidance stages of surgery. Currently,
only limited training is conducted, usually on unrealistic and expensive animal and
cadaver models. Surgery planning of port locations is also inadequate, often based only
on 2D images such as x-rays and angiograms. Finally, the current surgical guidance
method is imperfect, due to the small field-of-view (FOV) of the endoscope, and the
possibility that this view may become obstructed by blood, anatomy, moisture, etc.
The lack of proper 3D surgery training, planning, and guidance can lead to improper
patient selection, sub-optimal port placement, longer procedures, and increased risks
to the patient.
To address these issues, we want to develop interactive catheter and ablation guidance
strategies for the treatment of AF through:

- Real-time, MRI based navigation systems that guide catheter placement and then
  monitor lesion formation.

In other words, we need to generate a pre-operative 4-D model of the heart so the
surgeon has an anatomy reference of the heart that will help him navigate with the
catheter in the closed thoracic cavity. This problem can be divided in two major stages,
model creation from pre-operative data and the operative alignment of the model with
the real heart in the operative room. Our approach will consist in:

**Pre-operative stage:**

1. Find the boundary of the heart in a dataset of 4D- MRI images with minimal
   user intervention.

2. Render the surface that will model the beating heart.

3. Parameterize this set of volumes, i.e. describe the set of volumes as a set of co-
   efficients or numbers.

**Operative stage:**

4. Find the boundary of the 2D + time imaging that we have available during
   surgery.

5. Parameterize these boundaries as we did with the 4D model.
6. Find the exact location of the 2D images in the 4D model, i.e. we need to register the operative imaging with the pre-operative model.

7. As we will have the parameterized 4D model on one hand and the parameterized contours on the other, we will only have to compare this set of parameters and thus the computational cost will be reduced.

If we are able to solve this problem the surgeon will be able to see the virtual computer reproduction of the catheter and the heart model consistent with the real position of the catheter with respect to the heart.

The work presented in this paper presents a novel approach to solve the first three steps, segmentation and surface reconstruction from pre-operative cardiac MRI datasets. However, the segmentation and tracking algorithms applied on 1 will also be applied in steps 4 and 5. This will serve as a first step to find the position of the catheter with respect to the heart. So there are two major challenges, the first one is the segmentation and tracking of the heart and the second one is the 2D-3D registration to align the model with the operative environment. The focus of this thesis is on the segmentation of the heart, however we will also outline the current methods in registration of medical images because as we will see segmentation and registration are not two separate problems. In our case we bet for a registration based on surface to contour matching so a previous segmentation is necessary.

1.2 Previous Work in registration

To register the pre-operative model with the intra-operative images we will follow a geometric based registration approach. Geometric feature based approaches have the advantage of converging very fast which is necessary for surgical interventions. However, this methods rely on a previous segmentation. Other approaches have been proposed that do not rely on a previous segmentation. Consequently, we can classify registration methods in two main groups, geometric based and voxel similarity based.

1.2.1 Geometric Based Registration

1. *Point Based Registration:* This method consists in aligning two point sets usually defined manually in the skin of the patient or by landmark identification.
1. INTRODUCTION

Landmark identification in cardiac MRI is generally a very difficult task since there are a few spatially accurate anatomical landmarks. This landmarks are usually chosen to be the apex, the papillary muscles and the inferior junction of the right ventricle. This methods can be used as an initial registration for further optimization. Note that the registration of skin markers does not guarantee the registration of the heart since the position of the heart inside the body depends on the patient or respiratory contraction.

2. Registration methods based on heart surfaces:

The **chamfer matching method** \(^5\) is often used to register surfaces and point sets. In this method, the sum of the distances between the transformed points and a distance map built upon the segmented surface using the chamfer distance transformation is minimized. Also the iterative closest point algorithm (ICP) \(^4\) is used to elastically register surfaces and lines. In this approach, the shape to be registered is first placed in close proximity, in location and orientation, to the target shape \(^6\). Second, for each point on one shape, the closest point on the target shape is found and the distance between the two shapes is computed as a least square sum of their distances. Third, the pose parameters of the shape are adjusted incrementally at the direction which minimizes the sum of squared distances \(SSD\) between the two shapes. The process of calculating and minimizing this distance is iterated until convergence. This algorithm is effective, easy to implement, and robust if the initial pose is close enough to the true pose of the target shape. However, with a distant initial pose, the process can converge to local minima. The **"head and hat"** algorithm has also been proposed for registering medical images and was first presented to register brain images (12; 45). This algorithm models the contours from one of the images (usually higher resolution) as a surface (the "head") and the contours of the other image set as a series of points (the "hat"). The algorithm then determines the optimum rigid transformation, which minimizes the mean squared deviation between the points of the hat and the surfaces of the head by using the **Powell minimization algorithm**.
1.3 Previous Work in Cardiac Segmentation

1.2.2 Voxel similarity measures:

This class of methods have the advantage of not depending on a previous segmentation. However a strong correlation between pixels of different imaging sessions is assumed. Image registration methods can also be classified in terms of the type of search that is needed to compute the transformation between the two image domains. In search-based methods the effect of different image deformations is evaluated and compared. In direct methods, such as the Lucas Kanade method and phase-based methods, an estimate of the image deformation is computed from local image statistics and is then used for updating the estimated image deformation between the two domains. For the comparison of the two images several measures have been used, sum of squared differences SSD, sum of absolute differences SAD or by correlation coefficients CC all of them leading to very similar performances. Mutual information has also been used as a similarity measure. Mutual information theory measures the statistical dependencies between two random variables or the amount of information that one variable contains about the other. Mutual information can be qualitatively considered as a measure of how well one image explains the other. The mutual information is maximized at the optimal alignment.

For further reading on registration methods we refer to and specifically for cardiac analysis and segmentation and the references therein.

1.3 Previous Work in Cardiac Segmentation

The literature for segmenting cardiac medical images is vast. We will give a brief description of the previous work in cardiac segmentation and tracking. We have grouped the current cardiac segmentation in four groups, methods based on registration and optical flow, model based segmentation methods, region based approaches and finally active contour based methods.

1.3.1 Manual Segmentation and Registration

Manual segmentation requires human expert that selects a number of control points around the boundary of the heart and then this points are interpolated. This is a very tedious and time consuming work. Furthermore, we will need some post-processing if we want to find temporal and space consistency in our segmentations. Several authors
have proposed learning the cardiac motion by elastically registering a manually 3D-MRI segmented heart for the end-dyastolic phase to the rest of the cardiac phases \cite{28, 49, 51, 53, 54}, see figure. Therefore, the registration methods used are based on voxel similarity measures, see page \cite{4}, usually a bending penalty constraint is added in the minimization to find correspondences. This methods are prone to errors if the image sequences are noisy and the texture distribution change over time.

### 1.3.2 Model based

This methods rely on building a model statistically created from a database. The model is going to be deformed to wrap around the boundary of the particular unseen data that we want to segment. Model based approaches consist in two steps:

1. Model creation from the training set.

2. Model matching with unseen data

A very popular type of statistical models are Active Appearance Models, AAM \cite{8}. Recently AAM have become very popular for segmenting cardiac datasets \cite{7, 15, 47, 48, 57}. AAM models two features of the datasets, shape and texture. To model shape a set of landmarks in every dataset in the training set have to be identified. Landmarks are points that can be identified in all the datasets and thus are very useful to find correspondences. There are 4 kinds of landmark points:

\[ \text{Figure 1.1: Temporal registration} \text{- The end-dyastolic phase is elastically registered to successive phases}^{\text{[53]}} \]
1.3 Previous Work in Cardiac Segmentation

- **Anatomical landmarks** correspond between organisms in some biologically meaningful way. Such points are identified by a human expert.

- **Junctions** are points where different clearly distinguished boundaries meet.

- **Mathematical landmarks** points are located using some mathematical property such as high curvature.

- **Pseudo landmarks** points located between two landmarks.

The problem of automatically identifying landmarks is far from being solved, besides a landmark has to be identified in all the training sets so any landmark point in one training example has to define exactly the corresponding salient point as in all other training examples. This represents a challenge for human experts when doing the manual annotation because they have to be very precise.

Once all the training sets have been annotated, the coordinates of the landmarks each training set are arranged in a single vector. The order doesn’t matter as long as there is a 1-to-1 correspondence, by convention its usually $s = (x_1, y_1, z_1, x_2, y_2, z_2, ..., x_N, y_N, z_N)^T$ where $x_i, y_i, z_i$ are the coordinates of the $i$th landmark point and $N$ the number landmark points. After this we want to reduce the dimensionality and find a set of basis that with a few coefficients we can describe the main variations seen in the training set. This is precisely what principal component analysis (PCA) does. However, before applying PCA all the shapes in the training set must be aligned i.e. make all the shapes have the same rotation, translation and scale. The next step is to compile the individual (aligned) shapes $s_i$ of the training set to a matrix $S$ where $s_i$ is the $i$-th column vector. The last step in building the shape model then is to calculate the principal component analysis (PCA) of matrix $S$, \(1.1\)

$$s = \hat{s} + \sum_{n=1}^{k} c_n \Phi_n$$

(1.1)

Once we have the new basis, eigenvectors of $SS^T$, we can describe any shape as a linear combination of this set of basis. In \(1.1\) $s$ is an arbitrary heart shape, $c_n$ are called the modes of the shape model and $k$ is the number of eigenvectors used to describe the shape, so the greater the $k$ the more accurate the description. The greatness of PCA is that the first eigenvectors have almost all the information of the shape variations in
1. INTRODUCTION

the training set.

To model texture the process is very similar. Again the crucial step when obtaining the
texture is that we need a 1-to-1 correspondence among the training sets. To achieve this
first the Delaunay triangulation of the landmark points is performed and the normalized
grayvalue of the pixels inside the resulting triangles is stored in a single vector for each
dataset, \( g_i \). To combine both models the two vectors corresponding to shape, \( s_i \) and
texture \( g_i \) are concatenated in a single vector \( x_i \).

\[
x_i = \begin{bmatrix} s_i \\ g_i \end{bmatrix}
\]

\[ (1.2) \]

In fact this is not exactly true, the magnitude of both features is in general very
different and this leads to bad statistical results. To avoid this the data has first to be
normalized by some weight matrix. Once this is done, we PCA the data like in \[1.1\] and
we obtain the combined shape-texture model.

The second step is to match the model with the unseen data. Besides the \( c_n \) coefficients, rotation \( r \), scaling \( s \) and translation have to be identified. In practice, there are
two more parameters that have to be included in the matching process. The texture
is normalized with respect to mean and variance so this two parameters have to be
included, \( m \) and \( \sigma \). So the extended vector of parameters \( p = (c_n|r|s|t|m|\sigma) \) has to be
optimized to match the model with the data. The path to the right parameter values is
learned through the texture differences between the model and the data. So the texture
differences \( \delta g \) are going to tell us how the parameters should be changed \( \delta p \). Formally
an underlying function between these two parameters is assumed:

\[
\delta p = f(\delta g)
\]

\[ (1.3) \]

The function \( f \) in equation \[1.3\] is found by linear regression. This function \( f \) has
to be determined before doing the model matching. It is done obtaining enough pairs
\( \delta p, \delta g \) from the training sets. Usually for each dataset in the training set the parameter
values are changed starting from the right ones to obtain the error in texture differences.
This is usually the most time consuming stage of the method. Once the function
\( f \) is determined the search can be started. Other approaches have proposed integrating
the learned priors in a level set framework. [15] [16]
This method works very well when the database is big enough. However, this method fails when the heart shape is very different from all the heart shapes in the training set. To overcome this problem local deformations to fine tune the segmentation have also been used in [47]. Model based approaches require to have a huge amount of manually annotated datasets which is very time consuming and in many applications is very difficult to obtain enough datasets. Furthermore, when the database is very big the search can be very time consuming because the amount of texture samples is huge. One of the problems of AAM is that linearity is assumed and intuitively shape and texture is not linear. It’s like trying to model the appearance of a face as a linear combination of several other faces. However, this approach has gained popularity because exploits the prior information of the heart very well and always converges to a solution close to the target. There has also been people interested in identify automatically landmark points [7, 15, 20]. In [15] only geometric landmarks are identified and in [7, 20] the training sets are registered with respect to a reference image for the correct alignment and landmark localization although the training set is formed by 2D images.

1.3.3 Region based

This approach has been used in [33, 37]. This class of methods segment the different objects in the images by looking at specific features such as texture, intensity values or geometric features like shape and area. Usually the user selects several initial points and the features are learned from the vicinity and the region is grown or propagated until the features change more than a certain threshold. This method can work well with CT images like in [37] in which they use skeletons to segment 4D CT data. The main disadvantage of this approach when used with medical images is that the objects to segment, epicardium and endocardium are not as homogeneous as we would like. However, many people include region based terms in the search. In [33] they first use morphological operators and thresholding to keep the LV and RV in the images. Then, they merge the regions in adjacent slices, for the same volume and for different frames using similarity measures. Two regions in adjacent slices are likely to be part of the same object when both regions overlap, i.e. when the quotient between the intersection and the union of the two regions is close to one.
1. INTRODUCTION

1.3.4 Active Contour

Active Contour approaches, also known as snakes \cite{26} have been widely been used for segmentation of medical images \cite{4, 22, 24, 25, 30, 35, 49}. and the work presented in this thesis is based on active contours. The original formulation of snakes \cite{26} is based on a contour driven by two forces, an internal force and a external force, the first one will make the curve shrink and the second one will make it stop. It’s like stretching an elastic band and letting it go, it will shrink until it is stopped by the object that we are looking for. The basic idea is to minimize an energy functional depending on image features and the curve itself. Given a parameterized planar curve:

\[
C(p, t) = \begin{bmatrix} x(t, p) \\ y(t, p) \end{bmatrix}, \quad p \in [0, 1]
\]

is the parameterization and \(t\) is an artificial time, the energy:

\[
L(C, C_p, C_{pp}) = \int_0^1 \left( w_1 \|C_p\|^2 + \frac{1}{2} w_2 \|C_{pp}\|^2 + g(C) \right) dp \tag{1.4}
\]

is minimized. \(C_p, C_{pp}\) denote moments of order 1 and 2 respectively and \(g(C)\) is the image influence. The elasticity term, order of moment one, make the snake shrink to its center and the rigidity term, moment of order two, makes the curve smooth. The image influence is generally chosen to be:

\[
g(x, y) = \frac{1}{1 + \|G \ast \nabla I(x, y)\|^2} \tag{1.5}
\]

where \(G\) is a Gaussian filter and \(I\) is the image. The gradient descent solution for the Lagrangian \(L(C, C_p, C_{pp})\) \(\tag{1.4}\) is found by means of calculus of variations. The negative gradient solution is then:

\[
C_t = \frac{\partial}{\partial p} (w_1 C_p) - \frac{\partial^2}{\partial p^2} (w_2 C_{pp}) - \nabla g \tag{1.6}
\]

The problem of the original snake formulation is that is not geometric, i.e. the normals, curvatures and moments depend on the somewhat arbitrary parameterization. On the other hand, the geodesic active contours \(\tag{1.7}\), another curve-based segmentation method, is completely geometric. The basic idea is to minimize the length of weighted curve. The conformal factor is the image influence term of the original snake \(g(C)\) \(\tag{1.4}\)

\[
L(C, C_p) = \int_0^1 g(C) ds = \int_0^1 g(C) \|C_p\| dp \tag{1.7}
\]
The gradient descent solution is:

$$C_t = (gk - (\nabla g \cdot N))N$$  \hspace{1cm} (1.8)

We can see that the solution doesn’t depend on the parameterization, $k$ is the curvature and $N$ is the normal of the curve. It is illustrative to compare the energy terms of the snake formulation to the ones for geodesics. The term $gkN$ is the geometric analog of the elasticity term $\frac{\partial}{\partial t} \omega_l C_p$, it pushes harder the points with higher curvature. The term $\nabla g$ gets replaced by its projection onto $N$. Now the movement is restricted to the direction of the normal $N$.

For geometric representations of the active contours the Level Set representation of the curve has been commonly been used. The Level Sets method (40), instead of making the curve itself evolve controls a higher dimensional surface that has as a zero level set the contour. So the cross-section of the surface at the $XY$ plane is going to be the curve we want to control. Being the $C(s, t) : [0, L(t)) \mapsto \mathbb{R}^2$ a planar curve and $\phi(x, y, t) : \mathbb{R}^2 \times [0, T) \mapsto \mathbb{R}$ a function with zero level set:

$$C(t) := \phi(x, y, t) = 0$$  \hspace{1cm} (1.9)

One of the very good properties of level sets is that the dynamic equations that govern the contour can be easily implemented numerically. It has been proven that the evolution of a curve is only governed by the normal velocity to the curve, the tangential velocity doesn’t affect to the shape. It can easily be seen that the gradient of $\phi$ is parallel to the normal of the curve. The derivative of $\phi$ along the curve $C$ is zero because $\phi$ is constant and equal to zero along the curve, from that we derive that $\langle C_s, \nabla \phi(x, y) \rangle = 0$ and consequently the gradient must have the direction of the normal. The simplest evolution of a curve is:

$$C_t = \langle \vec{V}, \vec{N} \rangle \vec{N}$$  \hspace{1cm} (1.10)

Now let’s see how it’s related to the evolution of $\phi(x, y, t)$.

From the chain rule we obtain that the evolution of $\phi$ is:

$$\phi_t = \phi_x x_t + \phi_y y_t$$  \hspace{1cm} (1.11)
From here, and taking into account that the gradient of $\phi$ and the normal of the curve $N$ are parallel:

$$\phi_t = \langle \nabla \phi, C_t \rangle \quad (1.12)$$

$$= \langle \nabla \phi, V_N \vec{N} \rangle \quad (1.13)$$

$$= \langle \nabla \phi, V_N \frac{\nabla \phi}{||\nabla \phi||} \rangle \quad (1.14)$$

$$= V_N |\nabla| \phi \quad (1.15)$$

From that we can write the Eulerian formulation of the evolution of the curve,

$$\phi_t + V_N |\nabla| \phi = 0 \quad \phi_0 = \phi(x, y, 0) \quad (1.16)$$

The above equation 1.16 is the fundamental equation of level sets. So far it may seem crazy to handle a $2D$ problem by adding an artificial dimension, we are trading in a moving curve for a moving surface. However, topological changes are naturally handled. Furthermore, as we have seen the evolution of the curve can easily be controlled by taking into account the evolution of $\phi$. Better yet, the formulation can be extended to $3D$ with no change. The contour optimization problems using level sets always seek a solution that with the minimum curve length, the geodesic curve, satisfies certain conditions like the conformal factor in 4.1. The signed distance function is usually used to initialize $\phi(x, y)$ with $\phi_0$. The signed distance function is then a surface with height at $(x_i, y_i)$ equal the distance between the point $(x_i, y_i)$ and the contour. Define $\Omega_1, \Omega_2$ as the interior region and exterior region of the contour respectively, and a point $p \in \mathbb{R}^2$, the signed distance function is given by 1.17

$$\phi_0(x, y) = \begin{cases} 
0 & \text{if } p \in C(s) \\
-\|p, C(s)\| & \text{if } p \in \Omega_1 \\
\|p, C(s)\| & \text{if } p \in \Omega_2 
\end{cases} \quad (1.17)$$

Note in equation 1.17 that the signed distance of $C(s)$ always has the contour itself as the zero level set. In addition, the gradient is equal to one which is convenient for numerical properties.

Several active contour methods have been proposed for segmentation and tracking that differ from the two described above in the influence terms, the curve representation and the dynamics attached to the curve. We will give a brief description of the proposed methods:
1.3 Previous Work in Cardiac Segmentation

- **Curve representation:** The curve has been parameterized using splines (30), PCA, Fourier Series Expansion or in the simplest possible case by a piecewise linear approximation (particle-based). In this last case a set of control points are tracked over time (31). Purely geometric curves have also been used for cardiac segmentation (3). Parameterizing a curve introduces a bias, i.e. the shape is assumed to lie in a certain class. This can be a problem if the object we are tracking elastically deforms completely, but this is not the case for our problem. Level set methods have also been used for tracking medical data, usually a shape, dynamics and intensity priors are used (15, 16, 27, 41, 44). This methods are also known as Active Shape Models and are strongly related to the AAP. However, since the topology of the heart doesn’t change too much this method is generally sub-optimal when used with no prior constraints. The Fourier Series Expansion is generally the best parameterization because of the elongated shape of the heart.

- **Curve influence terms:** Modifications of the two forces that drive the contour have been proposed (3, 22, 25). Image features like, image gradient, optical flow and region based terms which characterize statistical properties inside and outside the heart have been used. None of them alone works well enough, image gradient in cardiac MRI images is noisy and struggles with boundary occlusion, i.e. when other organs overlap the heart and the boundary is not visible. Optical flow is not reliable in noisy MRI images and region based terms don’t work well enough because of the inhomogeneities of the heart. However, combinations of them can be used to drive the contour to the boundary.

- **Dynamical models:**

  Time dependence is a valuable source of information and it is used in most of the recent approaches. The motion model can be finite dimensional or infinite dimensional. Finite dimensional groups are usually chosen to be affine transformations, extending this groups to elastic deformations is generally not straightforward because the observations tend to be non-linear. Infinite dimensional groups allow any kind of deformation, i.e. no kind of dynamic motion is assumed. Since finite dimensional models can not account for elastic deformations a static optimization step is required, the correction step. Purely dynamical approaches have been presented to avoid this optimization step (38, 43, 50). In (50) they address the
1. INTRODUCTION

artificial separation of prediction and segmentation by representing the curve as a set of pre-specified points and each of them has an associated velocity. The idea is that the curve should stop when the potential energy is big enough to stop the kinetic energy, so it’s similar to the physical problem. The potential energy of the curve $U$ is the energy of the original snake formulation (26):

$$U = \int_0^1 \frac{1}{2} w_1 \| C_p \|^2 + \frac{1}{2} w_2 \| C_{pp} \|^2 + g(C) \, dp \quad (1.18)$$

The kinetic energy is the velocity or inertia of the curve $T$:

$$T = \int_0^1 \frac{1}{2} \mu \| C_t \| \, dp \quad (1.19)$$

The Lagrangian is then:

$$L = T - U = \int_0^1 \frac{1}{2} \mu \| C_t \| \, dp - \frac{1}{2} w_1 \| C_p \|^2 - \frac{1}{2} w_2 \| C_{pp} \|^2 - g(C) \, dp \quad (1.20)$$

And we can find the candidate minimizer of the above Lagrangian computing the first variational $\delta L$:

$$\mu C_{tt} = \frac{\partial}{\partial p} (w_1 C_p) - \frac{\partial^2}{\partial p^2} (w_2 C_{pp}) - \nabla g \quad (1.21)$$

In (43) they extend the formulation proposed by (50) formulating the same equation with a pure geometric representation of the curve and thus overcoming the parameterization drawbacks. Active contours are generally defined for planar curves, recently extensions to 3D active surfaces have also been used for segmenting the LV surface (24).
Chapter 2

Aims of the project

2.1 Final aim

Our ultimate goal is to develop interactive catheter and ablation guidance strategies for the treatment of Atrial Fibrillation through real time MRI based navigation systems that guide catheter placement and then monitor lesion formation.

2.2 Preliminary aims

Our initial goal though is going to be the first step of the big problem. We are going to provide the tools to semi-automatically segment the 4D MRI cardiac datasets and visualize the epicardium wall motion over the cardiac cycle. We want to segment the whole 4D dataset with minimal user intervention. To initialize the algorithm we will place some control points around the heart boundary for the first time instant. Then, we want to refine this rough segmentation and propagate it to segment the heart for the rest of the cardiac phases.
2. AIMS OF THE PROJECT
Chapter 3

Datasets

The 4D MRI datasets that we are going to work with consist of short axis SA images i.e. cross-sectional images of canine hearts transversal to its major axis. For each time instant we have one of this volumetric set of images. Our aim is to segment the heart in each of this time instants and slice locations to reconstruct a 4D model of the epicardium surface.

The MRI datasets acquisition parameters are the following:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Modality</th>
<th>XY res.</th>
<th>Z (inter slice res.)</th>
<th>temporal res.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset 1</td>
<td>MRI</td>
<td>1 mm x 1 mm</td>
<td>4 mm</td>
<td>15 cardiac cycles</td>
</tr>
<tr>
<td>Dataset 2</td>
<td>MRI</td>
<td>1 mm x 1 mm</td>
<td>6 mm</td>
<td>15 cardiac cycles</td>
</tr>
</tbody>
</table>

Table 3.1: dataset table - Dataset characteristics

The image acquisition is done by acquiring multiple successive slices for different times within the cardiac cycle. The acquisition is triggered by an ECG signal, and several cycles of the heart must be acquired. Several cardiac images corresponding to the same cardiac phase can be averaged to improve the PSNR \(^\text{(34)}\). The imaging is done on breath hold to avoid respiratory motion. Figure 3.1 illustrates the image acquisition process.

The first dataset was acquired after inducing ablation to the dog by radiation procedures. We can observe the lesion formed in the right ventricle of the heart. The second dataset was taken from a healthy heart, pre-ablation. In the figures below

\(^2\)Figure taken from "A review of Cardiac Image Registration Methods" \(^{32}\)
3. DATASETS

3.2, 3.3 we show a down-sampled version of the datasets. Along the horizontal axis we show the heart variation in time and along the vertical axis we can see it as a sliced volume. In the vertical direction the slices go from the base to the apex, most inferior part of the septum of the heart. In the horizontal direction we can appreciate that the heart is first contracting and then expanding.

In the second dataset we can clearly see in the top slices, three distinguished bright "chambers". This are the right ventricle tracts on the extremes and the aorta on the middle. Just looking at the data we can guess that one of the main difficulties to segment the heart for each of the time/slice location is the heart shape variation from top to bottom. In addition, all the veins coming out and in of the heart are going to be false contours we must avoid.
Figure 3.2: The figure shows various slices for different times and locations. This dataset includes some section of the atriums and all the ventricles to the apex. The bigger circular chamber with thicker walls is the left ventricle and the elongated chamber is the right ventricle.

Figure 3.3: Canine heart dataset2. The figure shows various slices for different times and locations.
Chapter 4

First Approach

4.1 Geodesic Curves to find the boundary

Before working with real data, we worked with a binary heart as a first approach. Start with the easier problem and climb step by step to the difficult problem. The binary heart was obtained from The Center for Cardiovascular Bioinformatics and Modeling. This is a simplification of our problem, the heart walls are white and the rest of the image is black. We will not extend too much with this first part, but we will outline just the most important ideas. We will want to find the epicardium boundary. To achieve this, we will start from an initial contour and will drag it to the heart outer wall. The contour is defined by a spline curve going through a sequence of control points. This is well known as Snakes. The Snake consists in applying 2 forces to the contour.

1. Internal Force : This force drives the curve to its center.

2. External Force : This force should be strong enough to compensate the Internal Force at the boundary location.

Our approach is going to be slightly different than the conventional snake algorithm. We know that for the real data, once we’ve found the contour for a pair \((t_i, z_i)\) we will use it as an initial guess for \((t_{i+1}, z_i)\). In other words we will start looking for the contour not very far away from it, so we won’t need more than two iterations to find the boundary. We will interpolate an initial set of points \(p_i\) with a spline. For each point in the curve \(C(s)\) we will look in the normal direction at the gray value of the
4. FIRST APPROACH

The Curve parameterization:

\[ C(s) = (x(s), y(s)) \quad s \in \{0, Curve\ Length\} \quad (4.1) \]

and the normal to the curve:

\[ \vec{N}(s) = \left( \frac{\partial x(s)}{\partial s}, \frac{\partial y(s)}{\partial s} \right) \quad (4.2) \]

The search lines correspond to the blue lines in figures 4.1,4.2 If this line of search crosses the contour we will find a big jump from 0 to 1 if we are outside the heart and 1 to 0 if we are inside. This big transition implies that there is going to be a high derivative at the contour location. So we will only look for high derivative values at each of the lines of search and will move the control points to the contour location. After doing this for all the control points we will interpolate them using splines to find the final contour. In case we have any outliers, points not belonging to the heart boundary, see figure 4.1 we will look for points in the spline with high curvature. We know there shouldn’t be irregularities because the heart is smooth so a point with a high curvature must be an outlier.

We will exclude all the points with higher curvature than a certain threshold, this can be seen in figure 4.2. The green line in figure 4.2 is the final contour after excluding the high curvature points. We will exclude the points over two times the mean of the curvature.

\[ K(s) = \frac{\| C''(s) \times C''''(s) \|}{\| C'(s) \|^3} \quad (4.3) \]

Equation 4.3 shows how we calculate the curvature. It tells us how similar to a circumference the curve is locally. We said we looked for the contour in the gradient of the curve direction, but we didn’t specify in which orientation. If we could determine if we are inside the region of interest, in our case the heart walls, we could know what orientation we should take. If we are inside the heart we should look outwards the curve and vice versa, if we are outside our region we should look inwards the curve. This can be seen in figure 4.3. In our simplified problem it is very easy to know if we are inside or outside. If the pixel in question is white we are inside and if the pixel is
4.1 Geodesic Curves to find the boundary

**Figure 4.1:** Outlier with high curvature

**Figure 4.2:** High curvature points are erased

black we are outside. When we work with real data we will need some estimation of the probability of a pixel belonging to a region. One possible approach would be to model the pdf distribution of the heart walls.

**Algorithm 1** Drive the contour to the boundary

1. Set of points $= p_1, \ldots, p_i, \ldots p_N$
2. Spline Interpolation to the points $\Rightarrow C(s) = p(s) = (x(s), y(s)) \ s \in \{0, C.Length\}$
3. **for** $s = 1$ to $N$ **do**
4. Find normal to the curve $\overrightarrow{N}(s) = \left( \frac{\partial x(s)}{\partial s}, \frac{\partial y(s)}{\partial s} \right)$
5. $LineSearch \equiv (x_i, y_i) = (x(s), y(s)) + \overrightarrow{N} \ast w_i$
6. Pick the pixels $r(i) = \{I(x_i, y_i)\} \ \forall x_i, y_i \in LineSearch$
7. Find the minimum derivative of the pixel’s grayvalues in the Line
8. $\text{argmin } r'(i)$
9. Move the point $p(s)$ to the new location $(x_i, y_i)$
10. **end for**
11. Remove points with curvature $K(s) \geq threshold$
4. FIRST APPROACH

4.2 Surface Rendering

We can use the algorithm \cite{1} described before to find the contour in each of the slice locations. We will start from the top and will successively descend to the bottom of the heart. Each boundary found will serve as an initial guess for the consecutive slice in the bottom level. Once we have all the contours, we would like to render a surface. To render the surface we need to connect the points of consecutive contours. There are many ways to address this problem, \textbf{Marching Cubes} \cite{29} is commonly used to render a 3D volume from a binary image volume, also Delaunay Triangulation is used to connect unstructured clouds of points. For the geometry of our problem we are going to use a much easier and so faster algorithm to connect the points. The algorithm is based on a simple triangulation of the cloud of points.
4.2 Surface Rendering

4.2.1 Triangulation

We have a set of contours, as many as slices has our binary heart. The contours are defined by piecewise polynomials, splines. First of all, we should evenly sample the contours to get a cloud of points. Second of all, we need to connect this points to obtain a wire-frame model of the heart. Third of all, we will give color to the faces of the wire-frame model. We used two different criteria to evenly sample the contours.

4.2.1.1 Curve sampling

A simple way to evenly sample the curve is to discretize the curve at points evenly spaced. The curve length between two consecutive points is going to be constant \( L \). If the curve length of each curve is \( L \) the sampling vector is going to be:

\[
\vec{s} = \begin{pmatrix}
0 \\
\frac{L}{N} \\
\cdot \\
\frac{L}{N} i \\
\cdot \\
L
\end{pmatrix}
\]

\[
C(\vec{s}) = x_i, y_i, z_i \quad Collection \ of \ points \quad (4.4)
\]

When we have sampled all the contours we have matrix of 3 coordinate points.

\[
P = \begin{pmatrix}
x_1 & y_1 & z_1 \\
\vdots & \vdots & \vdots \\
x_n & y_n & z_1 \\
\vdots & \vdots & \vdots \\
x_1 & y_1 & z_n \\
\vdots & \vdots & \vdots \\
x_n & y_n & z_n
\end{pmatrix}
\]

The problem with chordlength sampling though, is that particles travel along the curve when evolves and generally they don’t stay uniformly spaced, see figure see 4.4.

With this approach though, we need to align all the contours so the curve length parameterization \( s \) starts at the same location for each curve. We will see that is going to be more convenient for us to sample the contours with a different strategy. We will
4. FIRST APPROACH

Figure 4.4: Particle travelling—Figure shows how the particles travel as the curve evolves

sample each contour at $\theta = 5$ degrees separation with respect to the centroid of the contour.

$$Centroid = \frac{1}{N} \sum_{i=1}^{N} (x_i, y_i)$$  \hspace{1cm} (4.5)

To sample the spline at every 5 degrees we will first sample the spline at the curve lengths $s_i = L/N$, where $N = \frac{360}{\theta}$. Then for each $s_i$ we will take an interval $[s_i - \epsilon, s_i + \epsilon]$ and will evaluate the curve at 10 locations to get a set of points. We can see this points as vectors with its origin at the centroid, $v_{ij} = p_{ij} - \text{centroid}$, and pick the vector that is closer to the angle that we want $\theta_i$. Actually, we will take the vector with maximum projection to the unitary vector with angle $\theta_i$ $(\sin(\theta_i), \cos(\theta_i))$.

$$\text{argmin}_j v_{ij} \cdot (\sin(\theta_i), \cos(\theta_i))^T$$  \hspace{1cm} (4.6)

4.2.1.2 Specifying node connectivity

We want to connect the points to form faces. Usually, 3 or 4 vertex faces are used, we will use 4 vertex faces. Generally, to describe a wireframe model two matrices are needed, the matrix of vertices and the adjacency matrix which specifies how to connect this vertices. We will construct the faces the following way:

Let’s denote the $ith$ point and slice location $z$ as $p_i^z$. We want to sort the points in a matrix such that their four connected neighbors are the ”closest” points in the top and
4.2 Surface Rendering

bottom slice, left and right node in the contour respectively.

\[
\begin{bmatrix}
    p_{i+1}^z \\
    p_{i-1}^z \\
    p_i^z
\end{bmatrix}
\begin{array}{c}
\mid \\
\mid \\
\mid \\
\end{array}
\begin{bmatrix}
    p_i^z \\
    p_i^z \\
    p_i^z
\end{bmatrix}
\begin{array}{c}
\mid \\
\mid \\
\mid \\
\end{array}
\begin{bmatrix}
    p_{i+1}^z \\
    p_{i-1}^z \\
    p_i^z
\end{bmatrix}
\] 

(4.7)

With this matrix arrangement, we don’t need the typical adjacency matrix that specifies how the nodes are connected because of the implicit structure of the matrix. We will arrange the three components of the points \((x, y, z)\) in three matrices \(X, Y, Z\) and will use the \texttt{surf(X,Y,Z)} \text{matlab} command to render the surface. If the result doesn’t strike us as well enough there is still something we can do to obtain a finer mesh. For the points in the top contour, we will perturb its connected neighbors along the contour in the slice below until the Euclidean distance between the two points is minimum. The points can move along the contour but can not cross each other, so the line of movement is delimited by the neighbor points in the same contour. In other words, the point \(p_i^z\) can move within the interval \([s_i - \epsilon, s_i + \epsilon]\) to minimize the distance with its neighbor \(p_{i+1}^z\). The figure 4.5 shows the wireframe surface of the heart with 4 vertexed faces.

![Figure 4.5: Wireframe model of the binary heart](image)

The surface was refined following the process described above but we also smoothed the surface as we will explain on the next chapter.
4. FIRST APPROACH
Chapter 5

Epicardium segmentation

5.1 Understanding the data

Before explaining how to proceed to segment the 4D data it’s going to be useful to know what challenges we will meet with. Knowing how the features of the data behave will let us have a better understanding of the parameters, algorithms and strategies we will follow to achieve our aims. We will see that classical segmenting approaches like creating a probability model of the heart walls region, classical edge detectors or optical flow and registration schemes will fail to track/segment the heart. Looking for correspondences between adjacent cardiac phases is an extremely difficult job and consequently computing the optical flow to model the heart motion is not a trivial problem. To find correspondences we could use well known techniques like block matching (SAD, SSD, CC). Doing so, will fail to find the contour in $t + 1$ because the heart boundary grayvalues and texture change substantially from frame to frame as can be seen in figure 5.1. Some people use a combined edge and texture information to find the boundary of a volume. They construct a 2D histogram, edge value in one axis and gray value in the other axis and pick the pixels that fall into a certain bin. Usually, a bin with a high edge value and the desired grayvalue range. We can see in figures 6.15 and 5.2 that gray value not only changes in time but also within the same frame. So the grayvalue of the epicardium walls in our MRI images is neither constant in time nor in space and also changes from slice to slice.

As a consequence of this we conclude that texture is in general a not very reliable feature for the segmentation and tracking of the heart.
5. EPICARDIUM SEGMENTATION

**Figure 5.1: fig-Grayvalue** - The figure shows how the gray value at the boundary changes over time and space.

![Grayvalue](image1)

**Figure 5.2: RV fills up** - The figure shows how the gray value at the boundary changes over time and space.

![RV Fills Up](image2)

In figure 5.2 we see that the color changes because the right ventricle fills up and expands and we see a much brighter texture in that region. Another difficulty that we have to overcome is that there are a lot of veins and arteries coming out and in the heart. As we are only interested in the heart boundary the curve should our region of interest should exclude the arteries and veins. As we can see in figure 5.3 this is going to be a challenging work because the regions where the Vena Cave comes in the right atrium or the region where the aorta comes out the heart there is no edge at all. Thus, our algorithm should be capable of closing the contour preserving the overall shape even when there is no visible boundary. Another difficulty we will find is the fat stuck to the heart walls, this is going to be a real distractor when looking for the boundary. We can see the fat in figure 5.4. Finally, we will have to deal with our partner present in all engineering problems, noise.
5.2 Processing the data

5.2.1 Preprocessing

As we have seen in section 5.1 the MRI images are noisy and there are a lot of distractors like little veins that appear in the cross sections as bright spots around the contour. We need to clean the image and preserve the heart shape intact at the same time. To achieve this we will use morphological operators to erase the bright spots from the image. To erase the spots we will perform an opening using a disk shaped structuring element of radius 2 pixels se[n], see the result in figure 5.6. We will binarize the output from the opening and will use it as a mask B[n], see figure 5.6. So we will multiply the binary image with the original image pixel by pixel.

\[
B[n] = \gamma_{se}(I[n]) = (I[n] \Theta se[n]) \oplus se[n] \tag{5.1}
\]

\[
I_f[n] = B[n] \otimes I[n] \tag{5.2}
\]
5. EPICARDIUM SEGMENTATION

This way, we erase some of the bright spots in the image, preserving the heart shape and the grayvalue distribution inside the heart remains unchanged. See the final result in 5.8.

![Original image](image1.png) ![Output of opening](image2.png)

**Figure 5.5:** Original image  **Figure 5.6:** Output of opening

![Binary mask](image3.png) ![Final image](image4.png)

**Figure 5.7:** Binary mask  **Figure 5.8:** Final image

**Figure 5.9:** Steps to clean the images

5.2.2 Initial segmentation

The first step is to obtain a first estimation of the segmentation for the first cardiac phase to initialize the algorithm. This first segmentation has to be provided by the user.
5.2 Processing the data

The user will interactively select 6 control points close to the boundary for each slice for the first phase only. Then this points will be interpolated using cubic splines. We will see in the next sections that we will work with "cylindrical" coordinates because the radial representation of the contours will be more convenient for further processing. Cylindrical coordinates would be useful if a straight line can cross each of the contours close to its centroids. However, as we observe that contours for different slices are misaligned we will use a modification of the cylindrical coordinates. We will warp the major axis of the cylinder so its center line crosses the contours near its centroids, see figure 5.10.

![Curved cylindrical coordinates](image)

**Figure 5.10: Curved cylindrical coordinates** - The center axis goes by the centroids. The centroids for the first phase are used to parameterize all the phases.

First of all, we will sample the contours described with splines at 5 degrees of separation as described in section 5.2.3. Then we will find the centroids $c^z$ as:

$$c^z = \left( \frac{1}{N} \sum_{i=1}^{N} x_i^z, \frac{1}{N} \sum_{i=1}^{N} y_i^z, z \right)$$  \hspace{1cm} (5.3)

Once we have found the centroids for all the slices we will fit a 3D parabola to the points. This step is necessary to have all the centroids aligned. All the points of the surface will then be described with respect to this new centroids. Thereby, we will fit a second order polynomial to each of the 3 coordinates of the centroids 5.3.

$$t \rightarrow (x(t), y(t), z(t))$$  \hspace{1cm} (5.4)

$$t \rightarrow (p_1(t), p_2(t), p_3(t))$$  \hspace{1cm} (5.5)
5. EPICARDIUM SEGMENTATION

The parameter $t$ value for the point is chosen by Eugene Lee’s centripetal scheme, i.e., as accumulated square root of chord length \[5.6\] just as we did for the spline interpolation.

$$t(i) = \sum_{z=1}^{i} \sqrt{||c^z - c^{z-1}||}$$  \hspace{1cm} (5.6)

The formulation of the linear problem is the following:

$$
\begin{bmatrix}
1 & t_1 & t_1^2 & \cdots & t_1^n \\
1 & t_2 & t_2^2 & \cdots & t_2^n \\
1 & t_3 & t_3^2 & \cdots & t_3^n \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & t_m & t_m^2 & \cdots & t_m^n
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
\vdots \\
a_n
\end{bmatrix}
= 
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
\vdots \\
x_n
\end{bmatrix}
$$  \hspace{1cm} (5.7)

Matrix $V_{m\times n}$ is the Vandermonde matrix and has the form shown in the right hand side of equation \[5.7\]. The $t_i$ are the datasites, $a$ is a vector of polynomial coefficients and $x$ is the data. To find the polynomial coefficients the linear problem in \[5.8\] is solved in a Least Mean Squares fashion for each coordinate.

\[5.8\]

$$V_{ax} = x \quad V_{ay} = y \quad V_{az} = z$$

Figure 5.11: The blue dots in the figure correspond to the centroids and the red line is the LMS fit.
5.2 Processing the data

Now that we have found this new centroids that lie in a 1-dimensional smooth line we will find the polar coordinates of each of the contours using them as the centers.

5.2.3 Correct the boundary

Now that we have cleaned up the 4D dataset we can proceed to do the segmentation. The statement of the problem is the same as the one in chapter 4. We have an initial crude segmentation for the first frame and we want to refine it and propagate the result to the next frames. We will sample the curve at equally spaced angles with respect to the centroid, in our case we will pick a point at every 5 degrees \( \theta_i \) starting from north and going clockwise as shown in figure 5.12. See section 25 for more details.

![Figure 5.12: Sampling at 5 degrees separation](image)

The radial lines will be our search lines now. We will pick all the pixel locations that belong to the line of search. We know the contour is going to be relatively close to the initial guess so there is no need to look in the whole radial line, we just need to look within a window. We will use a window of 11 pixels, centered in the point of the curve.
5. EPICARDIUM SEGMENTATION

Figure 5.13: The red curve is the initial guess, the blue lines correspond to the radial search lines, and the curve on left is the grayvalue variation in one of the search lines.

For each radial line we will look at the grayvalue curve of the 11 pixels in the window, we hope the contour will be sharp enough to be detected by our algorithm. Looking for the heart edge is not an easy task even when we look for it locally as it is the case. In a normal scenario we would expect a jump in the boundary, i.e a high negative derivative if our search lines cross the heart from the inside to the outside. However, in some occasions the grayvalue curve is just the other way around, it jumps from a grayvalue to higher one, this is fairly common in the region where the heart is in contact with the chest of the dog. In some other occasions the border is blurry or noisy with a lot of high edges. Therefore, the derivative will not be enough. To overcome this problem we will first interpolate the grayvalues with a variational spline curve to find more precise curve features than applying direct discrete operators to the 11 point curve. We want to make sure we don’t move our points inside our region of interest, so we will look for the points where we find a knee of the second derivative. The knee of the second derivative indicates us a border because the curve is decreasing fast before the border and losing slope after the border. So we will pick the points where a knee of the second derivative is present as a possible candidates. The grayvalue curve as a function $y = f(x)$. We will then choose the candidate that minimizes a cost function build by five terms:

$$\argmin_{x \in \text{Window}} w_1 \frac{d^2 y}{dx^2} - w_2 \frac{dy}{dx} - w_3 y - w_4 |r(x, \theta_{i-1}) - r(x, \theta_i)| + w_5 \cos \alpha$$  \hspace{1cm} (5.9)
5.2 Processing the data

1. **Second derivative:** We want this term to be as high as possible

2. **First derivative:** We want this term to be as low as possible

3. **Grayvalue** In most of the cases the term should be low, although not always

4. **Smoothness term:** The radius of consecutive points in the curve shouldn’t change too much

5. **Image gradient:** The image gradient should be collinear with the normal direction of the contour. We approximate the normal with the radial direction.

The terms are controlled by the vector of weights \( w = (w_1, w_2, w_3, w_4, w_5) \) and all the features are normalized so they span \([0, 1]\). Let \( M, m \) be the maximum and the minimum of \( X \) respectively, to make \( X \) span \([0, 1]\) we will apply the following normalization:

\[
\hat{X} = \frac{X - m}{M - m}
\]  

(5.10)

We included the smoothness term in the pay function because we know the radius variation shouldn’t have discontinuities, so this term will penalize edge distractors like veins because this would make the radius make a jump. This term will help us find the contour, even when there is no contour, by following the shape trend of the previous curve points. With this cost function we don’t exclude completely the regions with positive derivative, we penalize them but they still have a chance to be selected. In

![Figure 5.14](image)

**Figure 5.14:** Figure shows how the boundary point is selected
5. EPICARDIUM SEGMENTATION

In figure 5.14 we see how the boundary location scores well in the first three terms of the cost function. The jumps of the third derivative (black line) will give us the location of the second derivative knees. So to find them we are actually looking at high fourth derivative values. We can also appreciate that in the border we see this slope decrease we explained before. If we just looked for high derivative values like most edge detectors do we would cut some pixels of the heart off because the slope hasn’t reached the bottom yet. The image gradient term is used to avoid local minima of the cost function. At the boundary of the heart the normal direction of the contour and the image gradient must be collinear. Since the heart shape is elongated its normal direction and the radial direction will be very similar. Therefore, we will use this radial lines to compare them with the image gradient. A high projection of the image gradient to the radial is a high indicative that we are at the boundary. For that reason we included the projection term $\cos \alpha$ into the cost function.

$$\cos \alpha = \frac{\vec{r} \cdot \vec{g}}{||\vec{r}|| \cdot ||\vec{g}||}$$

Equation 5.11 shows how the projection of the image gradient $g$ and the radial direction $r$ is computed. The image gradient $\vec{g}$ is found using the Sobel filter for the horizontal and vertical directions.

In figure 5.15 we can see that the red dots indicate the points where the boundary is not clearly identifiable. With this boundary strokes the best we can do is to interpolate the curve, so we want only the smoothing term to influence in the decision. Therefore, the point that minimizes the smoothing term will be chosen for that locations.

5.2.4 Curve parameterization and FFT

Once we’ve found all the boundary points we will obtain a rough curve around the heart boundary. We are looking locally at contour edges so although we have a smoothness term in our pay function, we obtain a curve with irregularities. There are 72 guesses for each slice-time, we can’t expect them all to be perfectly correct and congruent with each other. This is fairly common in these kind of problems. Humans find the boundaries of objects globally, we close the contour if necessary and visualize a smooth and continuous contour around the objects. Consequently, we will impose smoothness to this 72 guesses at each contour. Because of the elliptical shape of the heart we realized
that Fourier coefficients are well suited for reducing and constraining the dimensionality of our problem. We know the heart boundary must be smooth without jumps, so we should be able to express the contour as a sum of a few Fourier Coefficients, 4 or 5 maximum. If we needed more coefficients would mean that our curve is not correct, because only a lot of Fourier coefficients are needed to represent more complex curves.

So first of all, we will transform our coordinates, \((x_i, y_i)\) to \((r_i, \theta_i)\). We find out when we plot \(r_i(\theta_i)\) that is a noisy sum of sinusoids, see the plot on the left of figure 5.16.

(Actually it was after plotting radius versus angle that we thought it could be useful to us Fourier.) Second of all we find the first \(N_c\) FFT coefficients of the curve radius.

\[
R_k = \sum_{n=0}^{N-1} r_n e^{-\frac{2\pi i n k}{N}} \quad k = 0, \ldots, N_c - 1. \tag{5.12}
\]

Given that \(r_n\) are real:

\[
R_{N-k} = R_{*k}. \tag{5.13}
\]
5. EPICARDIUM SEGMENTATION

Figure 5.16: Shows the $r(\theta)$. The left plot shows the initial points and the plot in the right shows the guesses after keeping the first 5 harmonics.

We see in equation 5.12 that we only need to calculate the first $N_c$ Fourier Coefficients. As the $r_n$ are real we will find the phase of the coefficients with the complex conjugate pairs as shown in equation 5.13. So in total we will have the DC component plus $2 \times (N_c - 1)$ coefficients because we want amplitude and phase. Once we have the first $N_c$ harmonics we only need to go back to $(r, \theta)$ space as shown in equation 5.14 and from $(r, \theta)$ to $(x, y)$ again.

$$r_n = \frac{1}{N} \sum_{k=0}^{N_c-1} X_k e^{\frac{2\pi i}{N}kn} \quad n = 0, \ldots, N - 1$$  \hspace{1cm} (5.14)

Another way to see the whole process is the following: We have an initial irregular curve and will project it to a lower dimensional space. Before doing anything, our curve is represented by 72 points, we know the curve doesn’t have as many degrees of freedom so we will express the curve as a sum of elemental curves with a few coefficients. This elemental curves are sinusoids in $(r, \theta)$ space. Note that the phase of the sinusoids are equivalent to the rotation respect to its centroid in $(x, y)$ space and the DC component is just the scaling of the ellipsoid. If we take the pairs of Fourier coefficients and group them in its conjugate pairs $R_k, R_{N-k}$ we can see it as a single cosine function in which the modulus and the phase information are contained in the coefficients. So, basically we have a set of basis functions $b_1, \ldots, b_k$ in which $b_k$ is:

$$b_k = \begin{pmatrix}
1 \\
\cos w_k \\
\cos 2w_k \\
\vdots \\
\cos Nw_k
\end{pmatrix}$$  \hspace{1cm} (5.15)
5.2 Processing the data

We project our data to find the coordinates in this new subspace. The coordinates, the Fourier coefficients, are the phase and amplitude of the cosines basis functions $A_k, \phi_k$. In other words the coefficients will tell us the scaling and the rotation of the elliptical basis functions.

![Initial guesses coordinates in (R,Theta)](image)

**Figure 5.17:** Shows the $r(\theta)$. This plot shows how FFT is a good approximation and is not affected by the outliers

Other smoothing techniques like variational splines to approximate the points have also been tested and are highly sensitive to outliers and represent high curvature regions with difficulties.

So to sum up, our method to smooth the contour is the following:

\[
(x, y) \xrightarrow{xy \to \text{polar}} (r, \theta) \xrightarrow{\text{FFT}} (R_k) \xrightarrow{\text{IFFT}} (r, \theta) \xrightarrow{\text{polar} \to xy} (x, y)
\] (5.16)

In conclusion, the filtering of FFT coefficients will return us a smooth curve approximating our initial guesses and will get rid of the outliers. Furthermore, if we feel that 72 points is not enough we can interpolate the points in the curve using more points with the IFFT.
5. EPICARDIUM SEGMENTATION

Figure 5.18: Shows the \( r(\theta) \). The red points are the initial guesses and the blue points are after keeping the first FFT coefficients.

5.2.5 Propagate the contours

Up to this point we have explained how to drag an initial contour guess to the heart boundary. Once we have the boundary for the first time instant we will propagate it to the next frame. For now, let’s suppose we don’t make any assumptions on how the curve should change from frame to frame, we will see in next Chapter that we can estimate the velocities to find a better estimate of \( t + 1 \) with \( t \). So the contour in \( t - 1 \) will be our initial guess and we will corrected as explained before and propagate it to the next and so on until we have the contours for all cardiac phases, see figure 5.19.

5.2.6 Time smoothing

Once we have the contours for a complete cardiac cycle we observe jiggling in the motion. The jiggling is produced because the contours are found somehow independent of each other, i.e. we don’t find all the contours at the same time but one by one. So if we look at the node trajectories over time we see a jiggling that is clearly an artifact and not real heart motion. To avoid the jiggling we will fit a 5 coefficient polynomial \( P(t) \in \mathbb{P}_5 \) to the node trajectories in time. It is very difficult to show the effect of the smoothing without a video, but we could imagine that the ”time cylinder” formed by the successive stacking of all the temporal instants should be also smooth. We will find the polynomial
5.2 Processing the data

Fit in an LMS fashion, minimizing the squared distances between the polynomial and the points. As the heart motion is somehow periodic we will impose the polynomials the condition of being twice differentiable at the end data sites $P(x) \in C^2 \ x \in [1, T]$:

1. $P(1) = P(T) \quad P(t) \in \mathbb{P}_5$  
2. $\frac{dP(1)}{dt} = \frac{dP(T)}{dt}$  
3. $\frac{d^2P(1)}{dt^2} = \frac{d^2P(T)}{dt^2}$

Thereby we will fit our data with 3 degrees of freedom since we started off with 6 degrees of freedom and we spent 3 imposing the conditions in 5.19. All the polynomials in $\mathbb{P}_5$ that satisfy the equations in 5.19 will lie in the null space of matrix $B$. Matrix $B_{3 \times 6}$ has as many rows as the number of equations and as many columns as the number of coefficients. Being $a = (a_1 \ldots a_N)$ a vector of polynomial coefficients the problem that we have to solve is the following:

$$\min_a \|V a - r\|^2 \quad \text{given that} \quad a \in \ker(B)$$

Where $V$ is the Vandermonde matrix 5.7 and $r = (r_1 \ldots r_T)$ the radius of a node trajectory. The dimension of $\ker(B)$ equals the number of initial degrees of freedom $Dof = 6$ minus the number of constraints $N_c$, i.e. the degrees of freedom left for the

\[\text{Figure 5.19: The blue curve is the contour in } t - 1 \text{ and the red one is the corrected contour in } t\]
5. EPICARDIUM SEGMENTATION

fit. \( \dim(\ker(B)) = N = Dof - N_c = 3 \). The problem above in equation \(5.20\) can be solved by finding the solution directly in the null space of \(B\).

\[
Q \equiv \ker(B) = \{v_1, v_2, v_3\} \quad (5.21)
\]

\[
\min_{c_i} \| V(c_1v_1 + c_2v_2 + c_3v_3) - r \|^2 \quad (5.22)
\]

Which is equivalent to:

\[
\min_c \|VQc - r\|^2 \quad (5.24)
\]

Where \(Q\) is the matrix containing the orthogonal basis \(\{v_i\}\) of the null space of \(B\), any linear combination of this vectors will lie in the space of polynomials that satisfy equations \(5.19\). So we want to find the combination of this vectors that best fits our data \(5.23\). For that we will find the coordinates \(c_i\) with respect to the null space orthogonal basis \(\{v_i\}\). The well known solution of the above minimization is:

\[
c = (A^*A)^{-1}A^*r \quad \text{with} \quad A = VQ \quad (5.25)
\]

Once we have the coordinates in \(c_i\) of the null space of \(B\) we just have to go back to \(P_5\) space to find the vector \(a = (a_1 \ldots a_N)\) of polynomial coefficients.

\[
a = Bc = B((VQ)^*VQ)^{-1}(VQ)^*r \quad (5.26)
\]

**Figure 5.20:** Shows the node trajectories and the polynomial smoothing

We can see in figure \(5.20\) how the polynomial fitting erases the jiggling in time. What we see in the figure are the radius of the same node over time. The node \(n\) correspond to the point in the contour at the angle \(\theta_n\).
5.2 Processing the data

In figure 5.21 we can the surface made of the radius of each of the nodes for all the frames. Basically we are plotting $r(\theta, t)$. In this same figure we can observe the rough node trajectories before, figure 5.21, and after 5.22 much smoother. Note that there is a small variation of the radius over time, this is because the total heart volume only changes an 11% during a heart beat.

### 5.2.7 Velocity estimation

Up to this point we are propagating the contours without applying any velocity. When the heart is contracting or expanding the contour in $t - 1$ can be a very poor estimate of $t$. So if only we had some prior knowledge of how the heart boundary variation over time is, we could use it to estimate the contour trajectories. Unfortunately, the prior that the heart contracts and then expands is far too simple to be of any use in our problem. The heart variation over a cardiac cycle depends on the heart region. Our assumption is going to be that the heart walls variation has to be very similar for adjacent slices. If we can cope with this, we can find all the contours for a certain slice with no velocity prior and then estimate the velocity and apply it to the nodes of adjacent slices. We will proceed exactly the same way as before, the only difference now is that we will use a velocity to propagate the contours. Of course, in order to avoid error propagations we will correct after propagating the contour. Once we have all the contours for this new slice we will reestimate the velocities and propagate them to the next slice location.
5. EPICARDIUM SEGMENTATION

5.2.7.1 Velocity computation

The movement of the heart is defined by a matrix that we will call velocity matrix. The rows of this matrix are the velocities of each node and the columns are the velocities of a node at each temporal instant. So \( V(n,t) \) or \( V(\theta_i,t) \) if you prefer is determining how the node \( n \) has to move at time \( t \) to reach time \( t+1 \). As a first approach we calculated \( V(n,t) \) as a motion vector parallel to the normal of the curve at \( n \). This way to work out the velocities is computationally more expensive and leads to errors because it’s too sensitive to the curve slope. We will rather find the radial velocities since we already have the points evenly angled sampled.

\[
V(n,t) = \frac{r(n,t)}{r(n,t-1)} \quad (5.27)
\]

We will find \( V(n,t) \) as in equation \( 5.27 \). You may have appreciated that \( V(n,t) \) is not a velocity, is a scaling a-dimensional, the actual velocity would be calculated as \( V(n,t) = r(n,t) - r(n,t-1) \). We choose the scaling factor instead of the velocities because the scaling is scale invariant which in our case is desirable because we are propagating the velocities from one slice location to the above or below one. So, if we define the matrix of \( N \) nodes by \( T \) frames \( R \) with elements \( r(n,t) \), to find the vector of scalings \( v_i \) we just need to divide the row \( i+1 \) with the row \( i \). \( v_i \) is going to be the vector to go from the contour \( i \) to \( i+1 \). Thereby each of the \( T-1 \) rows of the matrix \( V \) will tell us how to move from time \( t \) to time \( t+1 \).

\[
V = \begin{pmatrix} v_1 & v_2 & \cdots & v_{T-1} \end{pmatrix} \quad (5.28)
\]

Observing the data we can differentiate three phases in a cardiac cycle, the sistole, when the heart is contracting, the diastole when the heart is expanding and a state in which the heart is relaxed. For example, for the second dataset for which we have 12 frames per cycle we can classify the frames in the following categories:

- Frames 12,1,2,3,4: Sistole
- Frames 5,6,7,8,9: Dilation
- Frames 9,10,11: Relaxed
5.2 Processing the data

According to this, the velocity matrix should describe this movement; if we look at equation 5.27 it is pretty straightforward to induce how the trajectory of node should be. When the heart is contracting the velocity scaling should be less than one, when the heart is expanding the velocity should be greater than one and when the heart is relaxed it should be equal to one.

\[
V(n, t) < 1 \implies \text{Heart Contracting} \quad (5.29)
\]

\[
V(n, t) < 1 \implies \text{Heart Expanding} \quad (5.30)
\]

Looking at equation 5.31 and taking into account the previous frame classification the plot of \( V(n_i, t) \) versus time should be similar to a parabola with its tip on frame 7 or 8. In figure we have plotted the surface of the whole velocity matrix, one axis are nodes or angles, and the other axis are time. If the heart contraction and dilation were perfectly uniform, the surface should be a parabola in the time and more or less constant in angle or node dimension through the nodes. Unfortunately, the heart mechanical motion is not that easy. We can see in figure 5.23 that some of the nodes trajectories follow the pattern described, but some of them follow more complex trajectories.

![Surface of the velocity matrix](image)

**Figure 5.23:** Surface of the velocity matrix

We have plotted in figure 5.23 the plane at one cutting the surfaces in two halves. Ideally the frames over the plane correspond to the diastolic phase and the frames below
5. EPICARDIUM SEGMENTATION

...correspond to the siastolic phase. The velocity matrix corresponds to a cross-section of the upper part of the ventricles.

5.2.8 The code

Algorithm 2 Find and track the heart boundary

1: Given a rough segmentation of the first cardiac phase, \( C_z^{(0)} \)
2: for \( z = 1 \) to \( N_{\text{slices}} \) do
3: \hspace{1em} for \( t = 1 \) to \( N_{\text{frames}} \) do
4: \hspace{2em} Find the estimate: \( \hat{C}_z^{(t-1)} \rightarrow C_z^{(t)} \)
5: \hspace{2em} Correct the estimated contour: \( \hat{C}_z^{(t)} \rightarrow C_z^{(t-1)} \)
6: \hspace{1em} end for
7: Smooth the node trajectories in time
8: Find \( V_z \)
9: Use \( V_z \) as an estimate of \( V_{z+1} \rightarrow \hat{V}_{z+1} = V_z \)
10: end for
Chapter 6

Heart Surface Reconstruction

6.1 M-Tensor Product Splines

Once we have segmented the heart for all the slices and frames, we want to visualize the 4D data as a surface representing the heart beat. We will use the algorithm used in section Surface Rendering to connect the points from all the different contour layers. As you may have expected the surface with no post-processing after the segmentation doesn’t look perfect. We have smoothed each single contour with elliptical Fourier filtering and time with polynomial fitting. The stacking of all the segmented contours is not smooth at all. We have found the contours at each slice location separately, so it’s not a surprise that the pieces of the puzzle don’t match perfectly. We know the heart is smooth in all the dimensions, x,y,z,t. So we have a cloud of 4D points \( p(x, y, z, t) \) and we want to fit a smooth hyper surface, a set of time surfaces, to them. Because of the shape of the heart it is going to be more convenient for us to work with cylindrical coordinates, spherical can work as well. The point is that instead of smoothing a d-valued function we will smooth a 1-valued function. To achieve that, we will consider the representation of the points as a radial function dependent on three parameters, angle, z height or slice location and time as shown in equation 6.1.

\[
R(\theta, z, t) = f(\theta, z, t) \tag{6.1}
\]

Remember that we have a point at every 5 degrees of each contour. So in total we have \( 72 \times N_{slices} \times N_{frames} \) points. Now each value of the function \( R(\theta, z, t) \) at \( (\theta_i, z_i, t_i) \) is the radius \( r_i \) of the point respect to the centroid of the contour at \( (z_i, t_i) \). The radius of
6. HEART SURFACE RECONSTRUCTION

the points with respect to its respective centroids shouldn’t change too much between adjacent points in any of the three dimensions.

Figure 6.1: We need to smooth the radius of the points in the 3 dimensions

So we will arrange this radius in a 3D matrix in a way such that all the points in the matrix have its physical neighbors as shown in 6.1 That means that the points are ordered in a matrix such that the real connectivity between them is preserved. The columns of the 3D matrix will be the contours for different slices and times. Starting with the first time-slice contour $C^{(t=1)}_{(z=1)} = r$ we will fill up the columns of the matrix with all the slice and time contours until $C^{(t=T)}_{(z=N)} = r$. Following the notation, $C_z^{(t)} = \{r_1, r_2, \ldots, r_n\}$ where each contour is a vector of radius now, the matrix would have the form shown in figure 6.2 We will use m-tensor product smoothing splines to smooth

Figure 6.2: 3D matrix structure
6.2 Smoothing splines

The statement of the problem is as it follows; we have a set of data points \( y = g(x) + n_i \) that follow a smooth function \( g(x) \) and are corrupted by some random unknown noise product of our algorithm \( n \). We are interested in recovering \( g(x) \) so we will project our data to a subspace of polynomial functions. We want to find a basis that can represent \( g(x) \) but can’t follow the fast noise. We choose piecewise cubic polynomial functions with some constraints to approximate our points. Given a set of points \( (x_i, g(x_i) + n_i) \) we would construct the following function:

\[
f(x) = P_i(x) \quad x_i \leq x < x_{i+1} \\
i \in 0, \ldots, n \quad P_i \in \mathbb{P}_3
\] (6.2)

The function \( f(x) \) minimizes the following expression:

\[
p \sum_{j=1}^{N} w(j)|y(j) - f(j)|^2 + (1 - p) \int |f''(x)|^2 dx
\] (6.4)

The first term of the cost function controls the accuracy of \( f(x) \) i.e. how close to the points we are, and the second term forces \( f(x) \) to be smooth. We have a trade off between accuracy and smoothness. The choice of \( p \) depends on which of this two conflict goals we accord the greater importance. For \( p = 1 \) we would obtain the least squares line fit to the data. Besides this constraint we will impose that \( f(x) \) is twicec
differentiable, $f(x) \in C^2$. This gives rise to the following set of equations:

\[ P_{i-1} = P_i x = g(x_i) \quad (6.5) \]
\[ P'_{i-1}(x_i) = P'_i(x_i) \quad (6.6) \]
\[ P''_{i-1}(x_i) = P''_i(x_i) \quad (6.7) \]
\[ (6.8) \]

At the break points of $f(x)$ the function, first and second derivative have to match. Polynomials $P_i$ may be expressed in Newton form:

\[ P_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3 \quad (6.9) \]

Inertially we have 4 degrees of freedom for each polynomial $a_i, b_i, c_i, d_i$, if we impose the smoothness constraints $6.6, 6.7, 6.8$ we only have one degree of freedom. Hence we will express all the polynomial coefficients as a function of $a_i$ and find the $a_i$ that minimize the cost function in equation $6.4$. Applying the continuity constraint $6.6$ to $6.9$ we find the following expression for $b_i$:

\[ b_i = \frac{\Delta a_i}{\Delta x_i} - c_i(\Delta x_i)^2 - d_i(\Delta x_i)^3 \quad (6.10) \]

With $\Delta x_i = x_{i+1}$ and $\Delta a_i = a_{i+1} - a_i$. Forcing the second derivative continuity $6.7$ gives us the following expression for $d_i$:

\[ c_i + 3d_i \Delta x_i = c_{i+1} \quad (6.11) \]
\[ d_i = \frac{1}{3\Delta x_i} (c_i + 1 - c_i) \quad (6.12) \]

Using $6.10$ and $6.12$ we can express $b_i$ as:

\[ b_i = \Delta x_i = x_i - \frac{2}{3} \Delta x_i c_i - \frac{1}{3} \Delta x_i c_{i+1} \quad (6.13) \]

Applying the second derivative continuity constraint $6.13$ gives us the equation:

\[ b_{i-1} + 2\Delta x_{i-1} c_{i-1} + 3(\Delta x_{i-1})^2 d_{i-1} = b_i \quad (6.14) \]
6.2 Smoothing splines

Using 6.12, 6.13 and simplifying we find the following relationship between the $c_i$ and the $a_i$:

$$\Delta x_{i-1}c_{i-1} + 2(\Delta x_{i-1} + \Delta x_i)c_{i-1} + \Delta x_i c_i + 1 = 3 \left( \frac{\Delta a_i}{\Delta x_i} - \frac{\Delta a_{i-1}}{\Delta x_{i-1}} \right) \quad (6.15)$$

Where $Rf$ is a symmetric triangular matrix with general row:

$$[\Delta x_{i-1}, \ 2(\Delta x_{i-1} + \Delta x_i), \ \Delta x_i]$$

We can write this relationship in a matrix form:

$$Rc = 3Q^Ta \quad (6.16)$$

and $Q^T$ the tridiagonal matrix with general row:

$$[1/\Delta x_{i-1}, \ -1/\Delta x_{i-1} - 1\Delta x_i, \ 1/\Delta x_i]$$

The relationship between $b$ and $a$ can also be written in matrix form:

$$b = Wa - Zc \quad (6.17)$$

where

$$W_{n \times n} = \begin{bmatrix}
-1/\Delta x_0 & 1/\Delta x_0 & \cdots & 0 \\
1/\Delta x_1 & -1/\Delta x_1 & \cdots & \\
\vdots & \ddots & \ddots & \\
0 & \cdots & -1/\Delta x_{n-2} & 1/\Delta x_{n-2} \\
& & \cdots & \Delta x_{n-2} \\
& & \cdots & \Delta x_{n-3} \\
& & \cdots & 2\Delta x_{n-3} \\
& & \cdots & 0
\end{bmatrix} \quad (6.18)$$

and

$$Z_{n \times n-2} = \frac{1}{3} \begin{bmatrix}
\Delta x_0 & \cdots & 0 \\
2\Delta x_1 & \Delta x_1 & \cdots \\
\vdots & \ddots & \ddots \\
2\Delta x_{n-3} & \Delta x_{n-3} & \cdots \\
0 & \cdots & -\Delta x_{n-2}
\end{bmatrix} \quad (6.19)$$

Using 6.16 we can finally write $b$ as a function of just $a$:

$$b = (W - 3ZR^{-1}Q^T)a = Fa \quad (6.20)$$

where $F = (W - 3ZR^{-1}Q^T)$
6. HEART SURFACE RECONSTRUCTION

Now, that we have found all the parameters in terms of $a$, we will construct the linear system of equation 6.4 and will find the solution in terms of $a$. Over each interval the smoothness term in 6.4 is the integral of the square of the line $(2c_i + 6d_i(x - x_i)^2)$ (second derivative of 6.9). Since for any straight line $l$:

$$\int_0^h l^2(x)dx = (h/3)(l^2(0) + l(0)l(h) + l^2(h)), \quad (6.22)$$

We can write the integral term as

$$(1 - p)\int_{x_0}^{x_{n-1}} |f''(x)|^2 = \frac{4}{3}(1 - p)\sum_{i=0}^{n} \Delta x_i (c_i + c_{i+1} + c_i^2 + 1) \quad (6.23)$$

And writing the cost function in 6.4

$$S = p(y - a)^T D^{-2}(y - a) + (1 - p)c^T Rc \quad (6.24)$$

expressing $c$ in terms of $a$ 6.16

$$S(a) = p(y - a)^T Dp^{-2}(y - a) + (1 - p)(R^{-1}Q^Ta)^T R(R^{-1}Q^Ta) \quad (6.26)$$

Because $D$ and $(R^{-1}Q^Ta)^T R(R^{-1}Q^Ta)$ are positive semidefinite. the cost function is minimized when $a$ satisfies

$$12(1 - p)(R^{-1}Q^T)^T R(R^{-1}Q^Ta) - 2pD^{-1}(y - a) = 0 \quad (6.27)$$

Isolating $a$

$$12(1 - p)(R^{-1}Q^T)^T R(R^{-1}Q^Ta) + 2pD^{-1}a = pD - 2y \quad (6.29)$$

and we solve the linear system

$$Ax = y \quad (6.31)$$

Once we know $a$ we can find $b, c, d$ with equations 6.17, 6.18, 6.12 respectively. For further details we refer to [9]
Finally, once we have the smoothing spline for each dimension we will take the m-tensor product of them.

\[ R(\theta, z, t) = f(\theta)g(z)h(t) \] (6.32)

This can be thought as a gridded cube such that its axis smoothly deform to get close to the data. Then the datasites are approximated by the geometric average of each of this curved axis that cross them. We can see the resulting radius of the nodes before and after the smoothing for a constant time \( t = cnt \) in figures 6.3 and 6.4, and for \( \theta = cnt \) in figures 6.5 and 6.6 respectively.

**Figure 6.3:** Radius 3D matrix cross-section for \( t = t_i \) before smoothing splines

**Figure 6.4:** Radius 3D matrix cross-section for \( t = t_i \) after smoothing splines

**Figure 6.5:** Radius 3D matrix cross-section for \( \theta = \theta_i \) before smoothing splines

**Figure 6.6:** Radius 3D matrix cross-section for \( \theta = \theta_i \) after smoothing

The 3-D radius matrix cross-section for a constant time can be regarded as the heart surface for a certain phase 6.4. Consequently we are looking at the parameterized heart
6. HEART SURFACE RECONSTRUCTION

surface. We can see in figure 6.4 how the radius go up and down as we go from the base of the heart to the apex advancing through the slices. When we move in the \( \theta \) direction we see the sinusoidal outline typical of elliptic shapes parameterized radially. In figure 6.6 when we travel in time it’s hard to appreciate the small radial contraction and expansion, this is due to the scaling of the picture which is almost 20 times bigger than the heart radial contraction. Once we have all the radius we go back to Cartesian coordinates to visualize the heart beat. We will obtain three matrices, one for each coordinate, for each phase of the heart. The elements of matrix \( X, Y, Z \) for a given time will be:

\[
\begin{align*}
\quad x_{ij} &= c_x^j + R(\theta_i, z_j) \sin(\theta_i) \\
\quad y_{ij} &= c_y^j + R(\theta_i, z_j) \cos(\theta_i) \\
\quad z_{ij} &= c_z^j
\end{align*}
\]

(6.33 - 6.35)

where \( c_x^j \) denotes the \( x \) coordinate of the centroid at slice \( j \). The matrices \( X, Y, Z \) preserve the structure of the nodes, in the columns they contain their successive contours. Thereby the connectivity of the nodes is implicitly derived from the structure of the three matrices. Each node will be connected to its neighbors in the matrix to find the wireframe of the heart surface.
Chapter 7

Results

In this section we will show some of the results we have obtained with our method. In figure 7.1 we can see the surface reconstructed from planar slices. In figure 7.2 we can see a rotated version of the mesh. Figures 7.17, 7.18, 7.16 show how the surface is reconstructed from the wireframe model of the epicardium, the results are shown with short and long axis\textsuperscript{1} cross-sections of the data volumes. It is almost impossible to show the motion of the epicardium without a video sequence since the outer wall moves very little, only a 10-12\% variation over the cardiac cycle.

Several segmentation results are shown in figure 7.3 for different slices and cardiac phases. We can see in the two upper rows how our method can close the contour where there is no edge. In the middle row we can see that the contour cuts the vena cava and only stiches to edges with radial image gradient. This is particularly useful to measure the radial motion and the volume. In the lower row the segmentation is also very difficult since the heart is overlapped by the thorax and the organs. Actually, this is one of the reasons we used second derivative features, to find the valleys between heart and overlapping regions.

In figure 7.4 we show some of the results obtained for the segmentation and tracking of the LV. We can see much more motion in the LV walls than in the epicardium. To adapt our algorithm to segment the LV we increased the search windows, since the motion is bigger, and we also increased the number of Fourier descriptors to represent

\textsuperscript{1}The Long Axis images in all the figures are obtained using cubic interpolation. Note that the original resolution of the dataset is 6 mm and the pictures have a resolution of 1 mm, the images are just obtained for visualization purposes.
7. RESULTS

Figure 7.1: Canine heart surface-Surface reconstruction

the more complex shape of the LV. The LV shape is complex for the mid-ventricular slices where we can see the papillary muscles and elastically deforms over the cardiac cycle more than the epicardium. However, it is easier to track because there is more contrast at the boundary. In addition, to track the LV we won’t have the deal with contour closing and occlusion. The LV surface also accounts for axial shortening motion during contraction. In the slices towards the apex we can see the LV disappear from the slices during contraction, this is because not only contracts about its minor axis but also about its long axis. To detect this kind of motion we compute the probability of the region belonging to the LV. The probability is computed based on mean intensity values and the region is delimited by the contour prediction from previous time. So we detect whether the LV appears in the slice or not. In case the LV is not detected the most inferior part of it is then assumed to lie between the last slice where it appears
and the next one. If we look at figure 7.4 we can clearly identify the phases of the heart, the first column correspond to mid-systole, the second one to end-systole, the third one mid-dyastole and the fourth one end-dyastole. We can see that for the systole column, the LV is at its maximum contraction and for the last one, end-dyastole it is at its maximum dilation, end-systole. Figure 7.5 shows the LV contraction for three differentiated phases in the cardiac cycle, mid-systole, end-systole, end-dyastole. The contraction is maximum for end-systole in the middle of the picture. We can also observe the papillary muscles indentations.

In figure 7.6 we show both contours for the epicardium and LV along with the surfaces together. You might appreciate the synchronous contraction of both surfaces.

Figures 7.7, 7.8, 7.9, 7.10 show the two surfaces together in different poses. This plots are taken from a video sequence to visualize the motion of the epicardium and LV together. We added transparency at the epicardium faces so the LV was visible. Figures 7.11, 7.12, 7.12 show the segmentation results for the LV by overlapping the surface over the data. We have plotted this long axis image because the LV is clearly visible, it is the bright chamber on the right of the picture, the chamber on the left.
7. RESULTS

Figure 7.3: Epicardium contours-The figure shows the result of the segmentation for different slices 6 9 16 in the vertical direction. The cardiac phases correspond to mid-systole, end-systole, mid-diastole, end-diastole.

hand side corresponds to the RV\(^1\). In figure 7.14 we can see how both surfaces perfectly fit the data. Figures 7.19 7.20 7.21 7.22 show more results for both surfaces along with short axis and long axis cross-sections, it is particularly interesting to see in figure 7.19 how the aorta comes out the LV blue surface.

\(^1\)Note that depending on the view the LV appears on the right and the RV on the left
Figure 7.4: Epicardium contours - Segmentation results for different crossections of the heart. The first three rows correspond to slices 6, 9, 16, the third row corresponds to the surface reconstruction of the LV. The cardiac phases correspond to mid-systole, end-systole, mid-dyastole, end-dyastole

Figure 7.5: LV surfaces - LV surfaces for different phases mid-systole, end-systole, end-dyastole
Figure 7.6: Epicardium and LV surfaces and contours - The figure shows the result of the segmentation for different slices 9 and 15 in the vertical direction. The cardiac phases correspond to mid-systole, end-systole, mid-diastole, end-diastole.
Figure 7.7: Epicardium wireframe in red and LV surface in blue

Figure 7.8: View1: Epicardium and LV surfaces

Figure 7.9: View2: Epicardium and LV surfaces

Figure 7.10: View3: Epicardium and LV surfaces
7. RESULTS

Figure 7.11: We can perfectly see the LV in this vertical cross-section

Figure 7.12: LV mesh along with the data

Figure 7.13: LV after adding faces to the mesh

Figure 7.14: Mesh of both the epicardium in red and the LV in blue
Figure 7.15: Long axis cross-section

Figure 7.16: Long axis cross-section along with the epicardium wireframe

Figure 7.17: 3D heart surface overlapped to the data

Figure 7.18: 3D heart surface from a different view
7. RESULTS

**Figure 7.19:** Long axis cross-section with LV surface in blue

**Figure 7.20:** Epicardium and LV surfaces, we can see how they fit the data for

**Figure 7.21:** Epicardium and LV wireframe models intersecting a short axis image

**Figure 7.22:** Epicardium wireframe and LV surface, we can clearly see how the aorta comes out the LV surface
Chapter 8

Discussion and Future Work

8.1 Conclusions

We have briefly presented a review of the current approaches for epicardium and endocardium segmentation and discussed the advantages and disadvantages of them. We shortly reviewed the anatomical background and outlined special properties of cardiac MRI data. The successfully achieved goals of this thesis are listed below:

- We have presented a method for segmenting the epicardium in 4D cardiac MRI datasets that exploits the spatial and time dependencies of the heart.

- Our method can track the epicardium with an error of less than 2 mm with very little user interaction. We have seen that the segmented surfaces are space-time consistent.

- We have adapted our method to segment the left ventricle with very little modifications obtaining acceptable results.

- We have presented the results obtained for the segmentation of the epicardium and the left ventricle. This surfaces have been displayed in cine sequences to visualize the dynamics of the epicardium and LV together.

- Both surfaces, epicardium and LV are time and space consistent with no irregularities.
8. DISCUSSION AND FUTURE WORK

Future work will be the validation and improvement of the method and extensions to make it suitable for cardiac surgery interventions. This will include the following issues:

8.2 Reducing the complexity

To obtain the final surfaces our methods goes through a series of steps. Each of this steps introduces a small undesirable bias, every data fitting step assumes the data lies within a certain class. Although we can not see this artifacts in our results, it can be a problem when the method is applied to other datasets. Therefore we will study the possibility of reducing the number of steps. In our algorithm we find all the contours for a certain slice location we smooth the time trajectories of the nodes and learn the dynamics from there. Instead we could jump the time trajectories smoothing step, learn the dynamics from the noisy data and smooth the resulting velocity matrix. Then the time inconsistent data is going to be smoothed in the final step with the time-space smoothing.

8.3 Initialization of the algorithm

In this thesis we have assumed that an initial rough segmentation estimate was given. We could make the process fully automatic if we find a first segmentation estimate close enough to the actual boundary. In they make use of the temporal correlation between images to initialize a snake. If we analyze a certain slice location and we look at the time variance in the pixel domain, we will see that the heart pixels will have in general a higher variance than background pixels. This is due to the wall motion of the heart during contraction and due to the heavy blood flow inside the heart.

\[
V_z = \frac{1}{T} \sum_{t=1}^{T} (I^t_z - M_z)^2 \tag{8.1}
\]

where \(M_z\) is the mean image:

\[
M_z = \frac{1}{T} \sum_{t=1}^{T} I^t_z \tag{8.2}
\]

We have experimentally proved that the variance is not discriminative enough to find a good initial guess for all the slices. Hence we will study the possibility of using
area and region based (33) similarity measures between adjacent variance images to find a consistent heart volume. This initial heart volume will serve as an initialization.

8.4 Shape priors

Our method makes no assumption about the heart shape. This is has the advantage of being general enough to work better with substantially different heart shapes in comparison with template based (55) or model based (7 8 23 47 48 57) approaches. However, our method struggles with the base and the apex of the heart. At the base the aorta occludes the boundary and at the apex the heart boundary is very difficult to detect even for a human eye. Therefore we will include shape priors to the method (13 20 27 44) for the correct segmentation of base and apex slices. We will include also shape priors to deal with the papillary muscle present in the left ventricular interior wall.

8.5 Include heart rotation and axial shortening to the model

Our algorithm models very well the radial movement of the heart. However the surfaces don’t model the heart rotation and twisting nor the axial shortening towards the apex. We plan to include this to movements in our segmentations. Right now our contours are parameterized using Fourier coefficients, so we have information about phase available. Note that a linear shift of \( \Phi \) in the phase of the coefficients represents a rotation in the contours of \( \Phi \) degrees. Thus we can easily find the \( \Phi \) that minimizes the distance between two consecutive slices. We can combine this with a minimization of the grayvalue sum of squared differences SSD at the node locations.

8.6 Respiratory motion

The pre-operative datasets where acquired using ECG-gated cine MR on breath hold. This requires to have the patients retain their breath during 15-20 seconds which makes
it implausible for intra-operative image acquisition. We want to extend our algorithm to be capable of compensating the moving artifacts due to the motion of respiration. Biplane coronary angiograms have been used to recover the 3-D displacements due to respiration using stereo reconstruction \cite{11,36}. A parametric motion model is used to separate cardiac and respiratory motion fields.

### 8.7 2D-3D geometric based registration

Our final goal is to register the intra-operative images with the pre-operative model. Due to patient’s body position and acquisition inaccuracies the pre-operative model is misaligned with the intra-operative images. Before doing the 2d-3D registration we need to align the coordinates of the model to those in the operating room. Therefore, a first registration using skin markers or by an ultra-sound based registration of the ribs of the patient to those in the model \cite{21} must be done first. Registration of coronary angiograms \cite{13} has also been proposed for model-to-operative environment alignment. Such registration step provides a suitable starting point for the 2D-3D registration. The time alignment will be done using an external marker, ECG signal. Then an efficient 2D-3D registration algorithm based on geometry features \cite{1} \cite{19} will be used. We want to make use of the desirable rotational properties of Fourier descriptors to register the 2D contour of the intra-operative images with the 3D surface model.
References

[1] 2d-3d registration based on shape matching. 70


REFERENCES


[19] Heng Huang, Contact Information, Li Shen3, Rong Zhang1, Fillia Makedon, Bruce Hettleman, and Justin Pearlman1. *Surface Alignment of 3D Spherical Harmonic
REFERENCES


REFERENCES


[38] Marc Niethammer, Allen Tannenbaum, , and Sigurd Angenent. Dynamic active contours for visual tracking. *IEEE TRANSACTIONS ON AUTOMATIC CONTROL*, 200X.


REFERENCES


[57] Sebastian Zambal. 3d active appearance models for segmentation of cardiac mri data. Master’s thesis.

Declaration

I herewith declare that I have produced this paper without the prohibited assistance of third parties and without making use of aids other than those specified; notions taken over directly or indirectly from other sources have been identified as such. This paper has not previously been presented in identical or similar form to any other foreign examination board.

The thesis work was conducted from 01/01/08 to 07/15/08 under the supervision of Dr. Dana H. Brooks and Dr. Gilead Tadmor at Northeastern University.

BOSTON,