Segmentation of the frozen region in MR images during cryo ablation in the liver

Master thesis

Siri Øyen Larsen

15th November 2006
Segmentation of the frozen region in MR images during cryo ablation in the liver

by Siri Øyen Larsen

Thesis presented for the degree of
Master of Science
in Computer Science and Engineering

Supervisors: Knut Mørken and Eigil Samset

Department of Mathematics
University of Oslo, Norway
November 2006
Aknowledgements

First of all, warm thanks to Professor Knut Mørken, my supervisor, for all the hours he has spent providing me invaluable help and comments in an encouraging manner. I am grateful to my co-supervisor Eigil Samset for sharing his experienced intuition, and especially for welcoming me to Harvard University and Brigham and Women’s Hospital in Boston, USA, where I stayed three months while I was working on this thesis. I also want to thank some of his co-workers in Boston who made my stay educationally inspiring, especially Daniel Kacher, who among other things gave me instructive comments during real-life cryo ablation surgeries that I was let to observe, and Raul San Jose for helpful advice on level set segmentation.

I would like to thank radiographer Terje Tillung at The Interventional Centre at Rikshospitalet for patiently guiding me through the experiments with the MRI scanner and cryo equipment. My appreciation also goes to my Dad, who proof-read an early draft of the text, and who repeatedly enlightens me about my English language.

Last, but not least, I thank my friends for making my life enjoyable and life worthy, inside and outside Blindern. And Mom, Dad, Heidi and Astrid - thank you for always giving me your love and support.

Siri Øyen Larsen
Oslo, November 2006
Contents

1 Introduction ................................................. 1
  1.1 Background ............................................ 2
    1.1.1 Cryoablation ...................................... 2
  1.2 Problem description ........................................ 4
    1.2.1 Segmentation process outline ......................... 6

2 Materials, experiments and image preprocessing ................. 9
  2.1 Magnetic Resonance Imaging (MRI) .......................... 9
    2.1.1 History ............................................ 9
    2.1.2 Scanner system and imaging technique ................. 10
  2.2 Experimental equipment ....................................... 12
    2.2.1 MRI scanning equipment .................................. 12
    2.2.2 Freezing equipment ........................................ 12
    2.2.3 Scan parameters ....................................... 13
  2.3 Preprocessing ............................................. 14

3 Level set methods .......................................... 15
  3.1 Why level set methods? .................................... 15
  3.2 Representation of the interface .............................. 16
    3.2.1 Level set representation ................................ 16
    3.2.2 Parametric representation ................................. 16
    3.2.3 Choice of representation .................................. 18
  3.3 Slicer .................................................. 19
  3.4 The level set equation ...................................... 19
    3.4.1 Solving the equation numerically ....................... 21
  3.5 The speed function ....................................... 23
    3.5.1 Smoothing term ...................................... 23
    3.5.2 Balloon term ....................................... 25
    3.5.3 Advection term ...................................... 28
    3.5.4 Termination of the interface evolution ................... 29
  3.6 Usage and results ........................................ 30
# CONTENTS

4 Phase unwrapping 35
   4.1 The phase of the complex MRI signal 35
   4.2 The general unwrapping procedure 37
   4.3 Our phase unwrapping method 40
      4.3.1 Algorithm outline 40
      4.3.2 Region growing 41
      4.3.3 Fringeline detection 44
   4.4 Results and discussion 45

5 Segmentation based on a phase strategy 51
   5.1 PRF temperature mapping 51
   5.2 Clipping the level set contour 52
   5.3 Phase profiles 54
   5.4 Results and discussion 58

6 Segmentation based on a curvature strategy in 2D 63
   6.1 Curvature profiles 63
   6.2 Results and discussion 67
      6.2.1 PRF images 71
      6.2.2 3D images 71

7 Segmentation based on a curvature strategy in 3D 79
   7.1 Surface curvature 79
      7.1.1 Mean, Gaussian and principal curvatures 80
      7.1.2 Numerical computation of curvature 83
   7.2 Surface curvature segmentation 83
      7.2.1 Watershed segmentation 84
   7.3 Results and discussion 91
   7.4 Future work 94
      7.4.1 Definition of the surface between ice and air 94

8 Summary and conclusion 101

Bibliography 107
Chapter 1

Introduction

Minimally invasive therapy uses imaging techniques to treat various medical illnesses, and its ideas have contributed extensively to improve the field of medicine over recent years. These techniques are made possible because of the great progress that has been made in developing medical imaging equipment. Various modalities, such as magnetic resonance imaging (MRI), computer tomography (CT), ultrasonography (US), and X-ray imaging, enable the surgeons and radiologists to diagnose, plan, and finally perform minimally invasive procedures.

From a mathematician’s point of view the ambition is to advance the technology assisting the surgeons in their work by developing computer programs that can better visualize the information contained in the images acquired previously or during the surgical procedure. This thesis provides a modest contribution to this field, more specifically the development of 3D models for temperature monitoring during cryo ablation of hepatic cancer tumours. Section 1.1 will briefly uncover the vast field of image guided minimally invasive therapy with a general introduction, followed by a more specific exploration of cryo ablation. The main objective of this thesis will be announced in section 1.2, and how we intend to pursue the problem will be outlined in section 1.2.1.

We begin chapter 2 with background material on MRI as an imaging modality in section 2.1, and next present how experiments were conducted and images acquired. In chapter 3 we study the theory of level set methods, a technique with a wide spectrum of application areas, amongst others segmentation of images. Chapter 4 deals with phase unwrapping - an operation that is required in order to utilize the so-called phase images from the MRI scanner. In chapter 5 we investigate whether the phase information can be exploited to improve thermometry and present both our method and the obtained results. We then study an alternative ap-
proach (with which we had more success) using geometric properties of the frozen area that evolves during cryo ablation. More specifically we present both some previous methods alongside a new, improved method utilizing curvature information at the border of the frozen tissue in order to complete the segmentation. The curvature approach is studied in chapters 6 and 7.

1.1 Background

Minimally invasive therapy has some obvious advantages over traditional treatment methods. The most obvious one includes that the surgeon is able to operate the patient making small incisions only, causing less post-operative trauma as compared to classical surgery. The time needed for recovery is minimized, and so are scars.

We have mentioned MRI, CT, US, and X-ray as being the most commonly applied imaging modalities assisting minimally invasive surgery. Which imaging modality to use depends upon the clinical situation. In general, the structures of the body which are involved in the operation will experience several deformations during the surgery, and intra-operative imaging is therefore an essential part of the surgery guidance.

The medical image data are usually displayed as a series of slices oriented in the plane in which they are acquired (called the acquisition plane). For the treatment to succeed it is crucial to be able to exploit as much information as possible from the image data. It may sometimes be difficult to understand the spatial relationship of the different organs and structures from the 2D-images alone. This motivates the work of constructing 3D-models of the imaged volume, which first requires some degree of image processing (segmentation etc.).

In this paper we will focus on one specific technique within minimally invasive therapy, guided by one specific imaging modality: cryoablation of hepatic cancer, and MRI, respectively.

1.1.1 Cryoablation

It often happens that surgical resection of the tumour is difficult or even impossible. In such cases cryoablation is an alternative. ¹ The technique is

¹In medicine, ablation in general means surgical removal of all or part of an organ, tissue, or structure. In addition to cryoablation, the resection can be done by surgery, hormones, drugs, radiofrequency, heat, or other methods.
1.1 Background

Figure 1.1: An open GE Signa MRI scanner.

performed with the patient lying in an open MRI scanner, enabling intra-operative image guidance. An open MRI scanner consists of two magnets, separated by a gap where the magnetic field is homogeneous, thereby providing access to the patient during scanning, see figure 1.1.

Cryoablation is a form of ablation where freezing is being used to induce cellular necrosis in pathologic tissue. The technique is performed by a surgeon inserting one or more cryo probes into the tumor, followed by argon gas being pumped into the probe, resulting in the gas expanding at the tip of the probe, and the temperature decreasing. When the area around the probe tip freezes, it leads to destruction of the cell membranes, which is lethal to the tissue. The body will later get rid of the destroyed tumor by itself.

In order to treat hepatic cancer by freezing the tumor, it is assumed that the tissue temperature must be reduced to $-40 ^\circ C$ to ensure that the cells die, [3]. It is recommended that all tumor tissue, and ideally an additional 1 cm thick rim of normal tissue surrounding the tumor, reach the target temperature. For this treatment to succeed, it is therefore indispensable to accurately evaluate the temperature in the area around the cryo

---

$^2$The exact value of the temperature needed to destroy malignant liver tissue has not yet been agreed upon, [16]. Tumor cell survival has been reported at temperatures lower than $-50 ^\circ C$ in certain types of tissue. However, temperatures lower than $-40 ^\circ C$ to $-50 ^\circ C$ are considered lethal to cells.
Cryoablation is also performed with US guidance. In fact, although cryoablation was explored as a method for treating liver tumors for several years, it was not extensively used until intraoperative ultrasonography became available at the end of the 1980s, see [16] and references therein. Later, MRI was explored as an alternative imaging modality. It has been shown that US techniques has some severe disadvantages, mainly caused by so-called acoustic shadowing, leading to image distortion, [8]. The use of MRI may facilitate this task. In MR images one may clearly distinguish the frozen region from the unfrozen tissue. Due to the very short T2* time of frozen tissue, the ice-ball appears as a region of near signal void (black areas in the image) by conventional MRI pulse-sequences. ³ In [30] Tacke et al. perform a study comparing US, CT and MRI for imaging of cryoablation in porcine liver, and conclude that MRI was found superior to the other modalities due to better ice/tissue contrast.

At the moment, cryoablation procedures are performed without intra-operative access to a program that can construct 3D models. The surgeon watches the screen of the MRI scanner and conducts the procedure based on knowledge obtained through experience in studying such 2D image series. Concern must be taken to ensure correct placement of the probes, prior to the freezing. It is essential to avoid critical structures such as aorta or the spleen - not only when placing the probes, but also during the development of the ice-ball. It is besides important to surveil that all of the target tissue becomes ablated, while at the same time harming as little of the normal tissue as possible. An automatic segmentation of the ice-ball during freezing would not only facilitate this task, but also standardize the method, and make it safer. As we will see in this thesis, image guided thermotherapy is an area still waiting to become conventionalized, and to achieve this goal, proper image processing techniques have to become well established.

1.2 Problem description

The fact that we need the temperature to become $-40^\circ$ throughout the tumor, means that it is important to determine the region where the tissue has reached this temperature, and not only to determine the border between frozen and non-frozen tissue [8]. In order to make up for the

³Some information about tissue with subzero temperatures may be obtained by using specialized MRI pulse-sequences, with ultra short echo times. Further explanation of MRI, including T2* time, echo time and pulse sequences, can be found in section 2.1.
1.2 Problem description

Figure 1.2: The border between frozen and non-frozen liver tissue is clear in MR images. Left: Liver before cryo therapy (freezing). The dark tubes are the metal probes already placed in the liver. Right: Liver after cryo therapy. The dark area within the liver parenchyma is frozen tissue.

lack of information in MRI, thermal models can be used to estimate temperatures within the frozen region. This way we may be able to locate the border of the cryo lesion which is the tissue that has actually become damaged as a result of freezing. (While the term ice-ball is used to describe the entire frozen region, the term cryo lesion is the region of the tissue that has reached the target temperature - a temperature lethal to the cells.) Samset et al. calculated estimated 3D temperature maps during cryoablation of liver by solving the simplified bio-heat differential equation, as first proposed by Hong et al. in [9], and they concluded that the accuracy of this simplified bio-heat model is acceptable, see [25]. However, this method requires that the border of the ice-ball is known, since it is needed as boundary condition in the differential equation. The border between frozen and non-frozen tissue is fairly easy to see in the MR images, since frozen tissue gives no signal in conventional MRI pulse sequences, as already mentioned. However, since also air is signal void in this setting, another problem arises in distinguishing air from frozen tissue, when the frozen region extends to the surface of the liver. Figure 1.2 clearly illustrates the situation in an in-vitro case in a bovine liver.

Previously the boundary provided as input to the temperature calculation algorithm has been found by manual segmentation by an expert/radiologist. 4 This paper will address the mentioned segmentation

4As far as we know, this method has not yet been used in any clinical situation. Present-day surgeries are conducted with guidance from non-segmented MR images only.
problem, and one of the main tasks will be the development of a method for solving it - one that requires little or no user interaction.

1.2.1 Segmentation process outline

Segmentation of an image is the process of dividing the image into different parts. In medical images the parts may constitute tissue from different types of organs, air, blood, and so on. When image guided surgery is conducted, parts of the image may include medical devices and tools, for example the cryo probes (metal) during ablation of tumors by freezing. We may attach a label to each pixel, indicating which category it belongs to. Sometimes we instead want to detect the boundaries of particular objects in the image, i.e. we must perform edge-based segmentation. Given for example an MR image, the segmentation process is usually more complicated than just localizing areas of the same grayscale depth, or edges due to intensity discontinuities. A common problem occurs when an image is disturbed by noise or other disruptive information, which may cause an edge to be blurred or even invisible, or conversely, an edge to be present in a location where there is no real border. Segmentation methods may be divided into two groups, based on what sort of information they rely on. While self-contained methods utilize information contained in the image itself, the model-based methods make use of a priori knowledge about the problem, [7]. The development of model-based methods is crucial when facing problems where borders can only partially be detected from the image.

In this thesis we confront the task of segmenting the border of the frozen tissue in a liver. We will divide the segmentation process into two main steps:

1. Segmentation of frozen tissue from non-frozen tissue.
2. Segmentation of frozen tissue from air.

Step 1 will be solved by implementing a preexisting algorithm known as level set segmentation, and is discussed in chapter 3. This method exploits information about the image intensity to locate the edges of the ice-ball. The result of this process is an interface separating unfrozen tissue (which has high signal intensity in MR images) from signal void areas.

In some images frozen tissue borders directly on air or other zero-signal areas. This happens where the ice-ball encounters the surface of the liver. Step 2 thus faces the challenge of drawing a border between frozen
1.2 Problem description

Figure 1.3: Result of level set segmentation in a multiple probe case. The interface consists of two separate parts. The level set segmentation is what we refer to as segmentation step 1. The level set interface in the image has grown out of the liver. This happens since both air and ice is signal void. Segmentation step 2 consists in locating points on the interface where it should be clipped, i.e. points where the interface starts to grow out of the frozen region in the liver and into the air above.

tissue and air - a border which is invisible in the images. It is therefore not possible to extract the boundary directly from the images, and instead we will use a model-based method. The boundary may be inferred from the level set interface, as we intend to do in step 2 of the segmentation process. More specifically we will clip the level set interface at locations where it grows out of the liver parenchyma, see figure 1.3.

To perform the second step we have tested two different theories, one of which is new. The new strategy attempts to exploit phase information from MR images to locate the border of interest. A thorough explanation will be given in chapter 5. The second theory is to use segmentation based on a curvature strategy, and derives inspiration from previous work done by Heuch et al. in [8]. For now, we will only reveal that the idea is based on the observation that the interface produced in the first segmentation step will often experience a significant change in curvature at the boundary between ice and air, and this property can be exploited for segmentation.
While we use level set methods in 3D (i.e. on the entire image volume) for step 1 of the segmentation, Heuch et al. use segmentation by snakes in 2D (i.e. separately on each image slice). Thus although the underlying idea of using curvature is the same, the case will be defined quite different with respect to interface representation, curvature calculation, and segmentation methods. Their approach, which operates on slice images in 2D, will be thoroughly explored in chapter 6. It will also be tested on our data. We next present a new method, where we extend the idea of using curvature information to three dimensions. Our curvature strategy in 3D is treated in chapter 7.

According to Heuch et al.’s article [8], the shape of the contour - having grown out of the frozen region - is dependent on the constellation of objects and/or empty space above the liver. In [8], the studied images were derived from experiments using a single cryo probe for freezing. In this thesis we must also face the concern of more complicated shapes due to multiple probes. When ice develops around each cryo probe, and when the probes enter the liver at different angles or somewhat apart, there may be two or three frozen areas gathering into one as the freezing process evolves. Unless the probes are placed close together, the interface that must be clipped may consist of more than one separate part, as illustrated in figure 1.3.

Heuch et al. perform the segmentation between frozen tissue and air by finding two clip points on the contour (in 2D) and then connecting the points by a line segment. Since we will also work with a surface in three dimensions, clipping must be done along a clip contour (still in 3D), which necessitates a discussion of how to define the missing part of the boundary of the frozen region, i.e. the part at the surface of the liver where the level set interface outside has been removed. These concerns will be addressed in chapter 7.
Chapter 2

Materials, experiments and image preprocessing

In this chapter we describe MRI as an imaging modality before we present the details around how it can be utilized for our experiments. We also derive a method for preprocessing the acquired images.

2.1 Magnetic Resonance Imaging (MRI)

Despite being a young technology, MRI has become established as an important and fast growing field of research. In the following section we will give a brief review of the history of MRI, before we move on to describing how it works as an imaging modality.

Much of this section is a review of material from [10], [29] and [23].

2.1.1 History

The MR phenomenon was first discovered in 1946, independently by two different scientists - Felix Bloch and Edward Purcell, for which both were awarded the Nobel Prize in physics in 1952. In the twenty year period that followed this discovery, various experiments were conducted using nuclear magnetic resonance (NMR) to study chemical and physical molecular analysis. The motivation for using MR to detect disease came in 1971, after Raymond Damadian had discovered that healthy tissue and tumors have different nuclear magnetic relaxation times. In 1973 Paul Lauterbur was the first to demonstrate magnetic resonance imaging, using a back-projection technique similar to that of computer tomography (CT), also introduced this year. The basis of current MRI techniques was laid by
Richard Ernst in 1975, as he proposed using phase and frequency encoding, and the Fourier transform. The first MR images of the body were taken in 1980, when a single image could be acquired in five minutes. Six years later the acquisition time per image was reduced to five seconds. Both Ernst and Lauterbur were awarded the Nobel Prize for their discoveries - in chemistry in 1991 and in medicine in 2003, respectively.

2.1.2 Scanner system and imaging technique

The essence of the MRI method is spin; a physical property of certain types of atoms, contained for instance in the body. The individual constituents of an atom (electrons, protons and neutrons), each possesses positive or negative spin. Particles with spins of opposite signs may pair up to wipe out the magnetic effect. In MRI, the most common way to obtain an image is to measure the tissue content of water. This can be done since the nucleus of a hydrogen atom contains an unpaired spin, making the water molecules possess the remarkable property of acting like small magnets. When these magnetic nuclei are placed inside an MRI scanner, the spins align with the magnetic field of the scanner, hereafter referred to as the longitudinal magnetization. The spins are next exposed to a radio frequency (RF) pulse matching a certain frequency depending on the type of atoms of interest. This will change the net magnetism of the tissue, as it becomes flipped into the transverse direction, where it starts rotating about the axis. The result is that two relaxation processes take place: first the spins will start to realign with the longitudinal direction of the outer magnetic field, and second, the net magnetization will start to dephase in the transverse direction. The reason for the latter relaxation process is that the spins which constitute the magnetic field experience slightly varying field strengths, and thus rotate at slightly different frequencies. Molecular interactions will also influence and cause this process. The dephasing thus depends upon the tissue type and its resonance frequency, the strength of the outer magnetic field, as well as the temperature state of the tissue. The more time that elapses, the larger becomes the phase difference. The decay of transverse magnetization caused by molecular interactions is said to be the pure $T_2$ effect. The $T_2$ time is the time this process would take if the magnetic field had been homogeneous. The time constant representing the combined process of decay of transverse magnetization (i.e. the process caused by an inhomogeneous magnetic field as well as the molecular interaction) is denoted by the name $T_2^*$ ($T_2$ star).

During the relaxation process, energy is lost from the dipoles to the
surroundings, and an RF signal is generated. Contrast in the image is created from measuring this signal, which is picked up by the receiver coils of the MRI system. The decoding of the signal is done by a computer using the inverse Fourier transform. The strength of the signal depends on the proton density in the tissue, and on the two relaxation times; T1 time describes the time it takes for the spins to realign with the longitudinal magnetization, and the T2 time is the time that the dephasing process takes. More specifically, T1 equals the time it takes to reduce the difference between the longitudinal magnetization and its equilibrium state by a factor of $e$, while T2 is the time to reduce the transverse magnetization by the same factor. These two relaxation processes occur simultaneously, with the restriction that T2 will always be shorter than T1.
We have not yet addressed how the MRI system specifies location of the image. So-called field gradients are coils that change the homogeneous magnetic field by setting up spatial linear fields. These spatial variations of the magnetic field are employed for slice-selection and spatial encoding. A pulse-sequence is created by mixing together the RF pulses and the gradients in an order with accurate timing. The pulse-sequence can be varied to prescribe the desired type of image. An example of a so-called spin-echo pulse-sequence is given in figure 2.1, which can be found in [23]. In a spin-echo pulse-sequence, the spins are first exposed to a $90^\circ$ degree RF pulse, flipping the direction of magnetism into the transverse direction. At some point in time after the $90^\circ$ pulse, a $180^\circ$ pulse is applied. The $180^\circ$ pulse causes the magnetization to at least partially rephase and to produce a signal called an echo. The echo time $TE$ is defined as the time between the $90^\circ$ pulse and the maximum amplitude in the echo signal.

Finally, an MRI system must contain a computer system, where pulse-sequences and parameters are loaded, and the electronics controlled during scanning. As soon as the signal has been acquired and decoded, the image can be reconstructed on the workstation and displayed.

2.2 Experimental equipment

Our experimental studies took place at The Interventional Centre at Rikshospitalet in Oslo. The sections below describe the type of experimental equipment that was used for imaging and freezing.

2.2.1 MRI scanning equipment

The experiments were performed in an open GE Signa SP (Special Procedures) MRI scanner with 0.5 Tesla field strength. The scanner is designed by General Electrical Medical Systems (Milwaukee, USA). It holds two magnets separated by a 60 cm gap where the magnetic field is homogeneous, thus providing access to the patient during scanning. Each of our experiments was performed on an in vitro bull liver placed on the patient bed.

2.2.2 Freezing equipment

Freezing was achieved using the cryosystem CryoHit designed by Galil Medical (Yokneam, Israel). The system is driven by pressurized Argon gas (300 bar) to cool down the tip of a probe. Gas is run through a central
2.2 Experimental equipment

control station to the cryoprobes. In the tip of each cryoprobe a Joule-Thompson heat motor brings the temperature down to -180 °C at a minimum. The temperature at the probe tip is continuously measured and controlled by utilizing a small thermocouple situated inside the probe tip. We used 2-3 cryo probes in each of the conducted experiments.

An optical traceable locator may be attached to the cryo probe in order to provide the coordinates of the probe tip location as input to the image processing that will be performed later on. We did not have access to such a locator during our experiments.

2.2.3 Scan parameters

MR images were obtained prior to and at the end of the freezing period. At each of the two points in time we acquired two sets of images, using two different pulse-sequences: 1

1. A 2D-FGRE (fast gradient echo) sequence. The image sequence is to be used for the phase strategy. We will refer to these images as PRF (Proton Resonance Frequency) images, cf. chapter 5.

2. A 3D-SPGR (Spoiled GRASS) sequence. The image sequence is to be used for the curvature strategy.

We conducted three experiments and will refer to them as Experiment 1, 2, and 3. Experiments 2 and 3 were performed using the same liver (one on each side of the liver) and parameters. Detailed parameters are presented in tables 2.1 and 2.2.

1With the exception of the first experiment where the 3D-SPGR sequence only was acquired.

### Table 2.1: Experiment 1

<table>
<thead>
<tr>
<th>Sequence</th>
<th>TE (ms)</th>
<th>Slice thickness (mm)</th>
<th>Spacing between slices (mm)</th>
<th>Pixel size (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D</td>
<td>15</td>
<td>1</td>
<td>0</td>
<td>0.9766×0.9766</td>
</tr>
</tbody>
</table>

### Table 2.2: Experiments 2 and 3

<table>
<thead>
<tr>
<th>Sequence</th>
<th>TE (ms)</th>
<th>Slice thickness (mm)</th>
<th>Spacing between slices (mm)</th>
<th>Pixel size (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRF</td>
<td>30</td>
<td>7</td>
<td>1</td>
<td>1.1719×1.1719</td>
</tr>
<tr>
<td>3D</td>
<td>7</td>
<td>1</td>
<td>0</td>
<td>0.9766×0.9766</td>
</tr>
</tbody>
</table>
2.3 Preprocessing

Before doing the level set segmentation we preprocess the images in order to remove the noise floor and facilitate the deformation of the contour. An area of signal void may still contain noise, and we wish to remove this noise by assigning zero value to all its pixels, such that the area appear totally black in the image.

We use a preprocessing method identical to the one that Heuch et al. used for preprocessing MR images for segmentation by snakes, see [8]. The preprocessing method consists of thresholding with a dynamically set threshold followed by median filtering.

We first set a temporary threshold $t_{\text{temp}}$ that reflects the level of intensity where $x\%$ of the pixels constitute the image background. The parameter $x$ should be chosen empirically and express a rough estimate of the fraction of pixels that is not part of any object in the image. Thus $t_{\text{temp}}$ is found by constructing an intensity histogram of the image and locate the intensity level that corresponds to the cumulative sum of pixels being $x\%$ of the total number of pixels. This type of thresholding is also called $p$-tile thresholding, as it assumes we know the percentile of a certain class of pixels. Typically one uses the preponderant class of pixels, which in our case is the background pixels.

The next step is to estimate the average tissue intensity. This is easily calculated as the average intensity of pixels falling above the temporary threshold. The final threshold $t_{\text{final}}$ is then set to $y\%$ of the estimated intensity value. The parameter $y$ must be selected by empirically as with $x$ above. We then perform thresholding of the image with the final threshold $t_{\text{final}}$ by setting all pixels below this value to zero.

Median filtering is a well known method to remove so-called salt and pepper noise from images. Thus the last step of the preprocessing consists of applying a median filter (of size $3 \times 3$ pixels) to our images.

Table 2.3 display a list of the $x$ and $y$ parameters used for the image data from the different experiments.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>PRF</th>
<th>3D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experiment</td>
<td>$x$</td>
<td>$y$</td>
</tr>
<tr>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>3</td>
<td>60</td>
<td>60</td>
</tr>
</tbody>
</table>
Chapter 3

Level set methods

The principal objective of this chapter is to show how level set methods can be applied in segmentation of medical images. We will both present a general introduction to the underlying mathematical theory and discuss how it can be used for the application on hand.

3.1 Why level set methods?

The aim of our work is to construct an automatic method for segmentation of the frozen area that develops during cryo ablation treatment. The first part of the segmentation process will consist of delineating the border between frozen and non-frozen tissue. This boundary can easily be recognized in MR images, as frozen tissue is signal void with this imaging modality. Separating the two kinds of tissue is a task suitably solved using level set segmentation methods, the reason for which should become clear in the current chapter.

Level set methods were first introduced by Osher and Sethian in 1988. They are numerical techniques for tracking evolving interfaces, and are conveniently used in a wide variety of applications. The level set methods handle well interfaces with sharp corners, cusps, topological changes, and three dimensional complications, [27].

When applying the level set idea to image segmentation, we are seeking to detect the boundaries of some object or area in the image that we want to segment. This is done by initializing an interface somewhere in the image, and then change it by letting appropriate forces act on it until it finds the correct boundaries in the image. For this reason we will be talking about an interface that is evolving in time. Level set methods stand out from other front-tracking methods in the way they represent the boundar-
ies, as they use an *implicit* representation of the interface. We will look at how this is done below.

### 3.2 Representation of the interface

#### 3.2.1 Level set representation

As mentioned above the level set method tracks an interface, usually a contour in two dimensions or a surface in three dimensions, although the general idea works in any dimension. But rather than describing the evolution of the interface itself, the level set approach operates on a function in one dimension higher. The interface is thus described as the *isocontour* of this function, hence it is an *implicit* representation. In order to model an evolving interface we let the level set function depend on time as well as space.

**Definition 3.1** Let \( \phi = \phi(\vec{x}, t) \) be the level set function. Then the interface \( \Gamma \), at a given point in time, \( t \), is given as the set of points in space that corresponds to the zero level isocontour of \( \phi \), i.e. \( \Gamma(t) = \{ \vec{x} : \phi(\vec{x}, t) = 0 \} \).

Figure 3.1 shows an example of a simple level set function of two variables. The level set function is a surface in 3D, while its isocontours are two-dimensional curves. The evolving interface may be described by allowing one of the level set function’s isocontours to represent it. Most commonly the zero isocontour is selected, as in definition 3.1.

Now, it should be justified why one might wish to “complicate” things with an extra dimension, instead of employing an *explicit* parametric representation. There are several reasons for this, as we will discuss next.

#### 3.2.2 Parametric representation

Traditional techniques for tracking interfaces include point based methods, where the interface is given through a discrete parametric representation. The spatial positions of points on the curve or surface at a given point in time are used to reconstruct the evolving boundary front by connecting them with line segments in 2D and (for instance) triangles in three

\[\text{1There is no specific reason for choosing the zero level isocontour, other than that this is merely a convention. One is free to choose the level that suits the problem. As an example, note that the zero level set of the function } \phi(x, y) = x^2 + y^2 - 1 \text{ and the } \phi = 1 \text{ isocontour of } \phi(x, y) = x^2 + y^2 \text{ both define the same contour, namely the unit circle.}\]
Figure 3.1: Level set representation of an interface at two different points in time. The level set function $\phi$ is shown on the left and the interface corresponding to $\phi = 0$ is plotted on the right. Note that $\phi$ is positive outside the interface, and negative inside. In the above example the interface first yield a single curve, while later in time the topology of the interface has changed - it has splitted into two separate curves.
dimensions, \[28\]. We thus have a representation of the contour given by values \( \vec{x}(s,t) \) at values \( s \in D \), where \( D \) is the space parameter domain, and \( t \) is the time parameter. The equation of motion of an interface \( \vec{x} \) is then given by a *Langrangian* formulation \(^2\)

\[
\frac{d\vec{x}}{dt} = \vec{V}(\vec{x}).
\]

(3.1)

Here \( \vec{V} \) defines the velocity of the points on the interface. One difficulty regarding this approach is that the connectivity of the points on the interface must be tracked, and is likely to change during the evolution. Another consideration is to ensure that the discretization is appropriately coarse at each point in time - it must be fine enough to reconstruct the interface, and at the same time, if the points come too close together, they may cross each other, or cause instability unless the time step is also adjusted accordingly. A remedy for such problems is to redistribute the points every few time steps, and add or remove points where this is necessary, i.e. reparameterize. However, this task quickly becomes quite complicated, especially in three dimensions.

A more serious problem arises if the evolution of the front leads to a change in its topology. The parametric approach is not capable of handling this in a general manner, unless special procedures are implemented for detecting possible splitting and merging of interfaces. This brings us to one of the main advantages of level set methods: since they represent the interface implicitly, they allow changes in topology in an elegant way. Moreover, while a method that uses an explicit representation must be modified when changing from two to three dimensions, the level set strategy works equally well in any dimension.

### 3.2.3 Choice of representation

In a previous study of the presented segmentation problem, Heuch et al. (\([8]\)) use parametrically defined *snakes* \(^3\) to represent the deformable contour. The techniques that they used restricted the problem to one probe.

\(^2\)In a *Langrangian* formulation the mesh is fixed to the contour, i.e. the discretization follows the contour. In an *Eularian* formulation on the other hand, the discretization is fixed in space, and the contour must flow through the mesh. The level set formulation is an example of the latter.

\(^3\)Heuch’s work is based on a discrete dynamic contour model described in \([15]\). *Snakes* was introduced by Kass, Witkin and Terzopoulos, see \([11]\). Other recommended literature about snake contour models include an article on gradient vector flow snakes by Xu and Prince, \([32]\).
3.3 Slicer

The background for our choice of segmentation method is that we wanted to deal with the most general form of the problem, i.e. when there are more than one probe, yielding a general shape of the frozen region. While the ice-ball usually takes the shape of a drop around one probe, with two or more probes the result will be more complicated. It starts out with an area of ice around each probe, and these areas will then grow into each other. The final shape depends on the constellation of the probes with respect to each other, as well as the duration of the freezing process. While Heuch et al. performed the segmentation in a slice by slice manner, we will run the evolution of the interface in three dimensions. This is easier to do with level set methods than with snakes.

3.3 Slicer

We have used an existing implementation of the level set method for segmentation, developed at the MIT Artificial Intelligence Lab and the Surgical Planning Lab at Brigham & Women’s Hospital, an affiliate of Harvard Medical School in Boston, USA. The level set module is incorporated in a program called Slicer, which is an open-source software environment for visualization, registration, segmentation, and quantification of medical data. Slicer is a software research tool. We have used it to load the DICOM images \(^4\) from the MRI scanner, perform level set segmentation, and generate 3D surface models for viewing the segmented structures. Further information about the Slicer software can be found on their website, www.slicer.org. The Slicer Level Set Module is documented in [12].

3.4 The level set equation

The level set function \(\phi : \Omega \times [0, \infty) \rightarrow \mathbb{R}\) is a scalar valued function of both space and time variables. Since we restrict our attention to the image segmentation problem, in space \(\phi\) is defined on the same rectangular domain as the image, \(\Omega \subset \mathbb{R}^n\). Usually we have \(n = 2\) (a single image) or \(n = 3\) (an image volume, i.e. a set of slice images). The level set function is initialized at time zero and evolves in time until it stops, hence the function’s time domain is \([0, \infty)\).

First of all we must build an initial value for the level set function, i.e. \(\phi(\vec{x}, t = 0)\). This is often done using a so-called signed distance function. If

\(^4\)DICOM stands for Digital Imaging and Communications in Medicine. It is a file format for storing medical images.
Figure 3.2: Initialization of the level set function as a signed distance function. The right image illustrates the signed Euclidian distance which is computed from the boundary as initialized on the left.

the boundary we are seeking lies in 2D/3D, we might initialize the front as a circle/sphere of small radius. We then construct the initial value

\[ \phi_0 = \phi(\vec{x}, t = 0) = \pm d, \]

where \( \pm d \) is the signed Euclidian distance from each point \( \vec{x} \in \Omega \) to the initial front - assigning a positive distance if the point lies outside the region bounded by the front, and negative if inside. For points that lie on the initial interface, the distance is of course zero, hence \( \phi = 0 \) here, as it should. See figure 3.2 for an illustration.

We can now describe the motion of the front by matching it with the zero isocontour of the level set function. As outlined in [28] we must always (as the interface evolves) make sure that the level set value of a point on the front with path \( \vec{x}(t) \) is zero, i.e.

\[ \phi(\vec{x}(t), t) = 0. \] (3.2)

Differentiating this equation with respect to time by the chain rule we obtain

\[ \phi_t + \nabla \phi(\vec{x}(t), t) \cdot \vec{x}'(t) = 0. \] (3.3)

Here \( \phi_t \) designates the partial derivative of \( \phi \) with respect to \( t \). Now, let \( F \) denote the speed that drives the evolution, more specifically, \( F \) is the speed in the outward normal direction to the level set interface. Then \( F = \vec{x}'(t) \cdot \vec{n}, \) where \( \vec{n} \) is the outward unit normal to the level sets of \( \phi \), hence \( \vec{n} = \nabla \phi / |\nabla \phi|. \) Manipulating equation 3.3 we get

\[ \phi_t + \frac{\nabla \phi}{|\nabla \phi|} \cdot \vec{x}'(t) |\nabla \phi| = 0 \]
3.4 The level set equation

or

\[ \phi_t + (\vec{x}'(t) \cdot \vec{n}) |\nabla \phi| = 0 \]

which brings us to the general form of the level set equation

\[ \phi_t + F|\nabla \phi| = 0 \tag{3.4} \]

given \( \phi(\vec{x}, 0) = \phi_0 \).

The beginning of this section dealt with how we can specify \( \phi_0 \) as the initial value of the level set function. In our case we will run the evolution of the levelset in three dimensions, thus we can initialize \( \phi_0 \) to be zero at one or several small spheres, each sphere being centered at the tip of a cryo probe.\(^5\) For the purpose of covering the rest of the function domain we compute \( \phi_0 \) at each grid point as the signed distance from this location to the closest point on the initial interface.

The goal of the speed function \( F \) is to act on the interface and “pull it” towards the edges of the image. We should therefore model the speed function in such a way that when the interface reaches the desired position, the speed becomes zero, hence \( \phi_t = 0 \) and the interface must stop. At the same time the speed function should affect \( \phi \) so as to make sure it is kept smooth. Clearly, the function \( F \) must be defined in a way suitable for the given application. In section 3.5 we give a detailed discussion of the implementation that we used for our task.

3.4.1 Solving the equation numerically

After initializing \( \phi \) and \( F \) at the grid points, the interface can be moved across the grid by evolving \( \phi \) forward in time by applying numerical methods to update its values. Let \( \phi^n = \phi(t^n) \) represent the values of \( \phi \) at a given point \( t^n \) in time. We must update \( \phi \) by finding new values after some time increment \( \Delta t \), i.e. finding \( \phi^{n+1} = \phi(t^{n+1}) \) where \( t^{n+1} = t^n + \Delta t \). This can be done using a simple first-order accurate method for the time discretization, the forward Euler method, \([19]\). We compute \( \phi^{n+1} \) by approximating \( \phi_t \) at time \( t_n \) as

\[ (\phi_t)^n = \frac{\phi^{n+1} - \phi^n}{\Delta t}, \tag{3.5} \]

which when substituted into equation 3.4 and rearranging terms gives

\[ \phi^{n+1} = \phi^n - \Delta t (F^n |\nabla \phi^n|), \tag{3.6} \]

\(^5\) The level set module in Slicer (cf. section 3.3) allows the user to pick seed points (called fiducial points in this context) interactively by pointing the mouse in the image, and also to choose the radius of the spheres centered at the fiducials.
where the superscripts in $F^n$ and $|\nabla \phi^n|$ denote that the respective functions are evaluated at time $t^n$. Care must be taken when deciding upon a finite difference scheme to approximate the space derivatives. Since we are using a finite computational grid we are furthermore required to implement boundary conditions. We refer to the literature for further details about the numerical solution of the level set equation. Both Osher & Fedkiw’s book [19] and Sethian’s book [28] give thorough explanations.

Efficiency

When the interface we are seeking, as defined in Definition 3.1, is an object in $\mathbb{R}^n$, we must resolve an $n$-dimensional set in space, which may be computationally expensive if we solve the level set scheme in the straightforward manner. Instead of updating the level set function in the entire computational domain, the task can be relieved in part by focusing on the area of the domain close to the isocontour - the rest of the space may be left unresolved, [19]. This local approach to discretizing implicit representations stores and updates only the values of the level set function $\phi$ at points in a neighbourhood of the zero level set. It is known as the Narrow Band approach (see [28]) since we update the level set function only in an area shaped as a narrow band around the zero level set. Points that are outside the narrow band are given large positive/negative values if they are outside/inside the front itself. Of course, since the front is moving, the narrow band must also be advanced together with the front. Instead of doing this every time step, we stop and rebuild the narrow band only when the zero level set has reached the edge of the band. To rebuild the band requires that we compute the signed distance to the zero level set, i.e. reinitialize the distance map. Keeping $\phi$ approximately equal to the signed distance (the accuracy of the approximation depending on how often we perform reinitialization) is also desirable for another purpose: it helps to avoid numerical instabilities and to retain a smooth function. Furthermore, maintaining $\phi$ a signed distance function may simplify certain computations of geometric formulas, such as the one for curvature, since we then have $|\nabla \phi| = 1$, [19].

---

6There are several variants of the Narrow Band method. Sethian proposes in his book [28] to use Fast Marching Methods for reinitializing, while the Slicer implementation uses a method described by Krissian and Westin in [13].
3.5 The speed function

In our application the speed function \( F \) in equation 3.4 is decomposed into three terms: a smoothing term \( F_s \), a balloon term \( F_b \), and an advection term \( F_a \), such that

\[
F = F_s + F_b + F_a. \quad (3.7)
\]

The function \( F \) is defined on the same domain as the level set function and the image, and changes together with \( \phi \) as the interface evolves. We will see that the terms of \( F \) are functions of the level set function \( \phi \) and the image intensity \( I \), which represents the inner and outer forces that act on the interface, respectively. The interpretation of the different components of the speed in equation 3.7 will be explained in the subsections to follow.

In order to understand how the interface develops with the speed function, we should look at what happens to points on the interface as they evolve in the normal direction. Remember that \( F \) denotes the speed that drives the evolution of the interface in the normal direction, i.e. \( F = \vec{x}'(t) \cdot \vec{n} \), where \( \vec{n} = \frac{\nabla \phi}{|\nabla \phi|} \) as before. Since points on the interface move in the normal direction, we have

\[
\vec{x}'(t) = \beta \vec{n}, \quad (3.8)
\]

for some scalar valued function \( \beta \). If \( \beta \) is positive, the points on the interface move in the outward normal direction, while if \( \beta \) is negative, they move in the inward normal direction. We may then derive

\[
F = \vec{x}'(t) \cdot \vec{n} = \beta \vec{n} \cdot \vec{n} = \beta,
\]

since \( \vec{n} \) is a unit vector. From this we conclude that the level set equation moves points on the interface (or generally points on any level set of \( \phi \)) in the outward normal direction by a quantity corresponding to \( F \) evaluated at that point. Alternatively: \( \phi_t + F |\nabla \phi| = 0 \) is equivalent to \( \vec{x}'(t) = F \vec{n} \). This statement will become useful in understanding how the speed function is defined and will be exploited below.

Most implementations of the level set method allow adjustment of the contribution that each of the terms of the speed function makes to the equation. This can also be done in Slicer by adjusting a coefficient for each of the speed function terms.

3.5.1 Smoothing term

As the name implies, this term controls the smoothness of the contour. The term involves computing the (scalar) curvature \( \kappa \) of the level set function
Level set methods

\[ \phi = 0 \]

interface

\[ \kappa < 0 \]

\[ \kappa > 0 \]

\[ \phi < 0 \]

inside

\[ \phi > 0 \]

outside

Figure 3.3: An example of a level set function in 2D. Left: Convex regions have positive curvature, while concave regions have negative curvature. Right: The red and blue arrows show in which direction the interface moves and indicate negative and positive curvature respectively.

\[ \phi, \] and thus represents an inner energy of the interface (it is a function of \( \phi \) alone).

It can be proved that motion under curvature reduces variation in the curve, yielding the wishful smoothing effect. \(^7\) This is also intuitive, as illustrated by the two dimensional case in figure 3.3. Imagine that each piece of the curve moves in the outward normal direction with a speed proportional to negative curvature, \(-\kappa\). Parts of the curve where the curvature is positive will thus experience negative speed and move inward, while negative curvature will yield outward motion.

When the level set function evolves in three dimensions (as in our case), one must make a choice of what type of curvature to use, e.g. Gaussian, mean or minimal curvature. The Slicer Level Set module gives a choice of mean or minimal curvature. The minimal curvature is meant especially for segmentation of small structures with high curvatures, or tube-like structures such as arteries. For our segmentation problem the mean curvature is well suited, since it tends to preserve the shape of more circular shaped objects.

The mean curvature \( \kappa_M \) may be shown to equal the divergence of the normal to the level set function, \([28]\). The divergence of a vector field \( F = (F_1, F_2, F_3) \) is defined as \( \text{div} \; F = \nabla \cdot F \), where \( \nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}) \). Using

\(^7\)For a proof of how motion under curvature decreases the total curvature, see for example [26].
3.5 The speed function

Subscripts to denote partial derivatives, we derive the mean curvature \( \kappa_M \) as

\[
\kappa_M = \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|},
\]

where \( |\nabla \phi| = \sqrt{\phi_x^2 + \phi_y^2 + \phi_z^2} \), yielding

\[
\kappa_M = \frac{\phi_{xx}|\nabla \phi| - \phi_x \frac{2\phi_x \phi_{xx} + 2\phi_y \phi_{yx} + 2\phi_z \phi_{zx}}{2|\nabla \phi|}}{|\nabla \phi|^2} + \frac{\phi_{yy}|\nabla \phi| - \phi_y \frac{2\phi_x \phi_{xy} + 2\phi_y \phi_{yy} + 2\phi_z \phi_{zy}}{2|\nabla \phi|}}{|\nabla \phi|^2} + \frac{\phi_{zz}|\nabla \phi| - \phi_z \frac{2\phi_x \phi_{xz} + 2\phi_y \phi_{yz} + 2\phi_z \phi_{zz}}{2|\nabla \phi|}}{|\nabla \phi|^2},
\]

which can be simplified to

\[
\kappa_M = \frac{\phi_x^2 (\phi_{yy} + \phi_{zz}) + \phi_y^2 (\phi_{xx} + \phi_{zz}) + \phi_z^2 (\phi_{xx} + \phi_{yy})}{|\nabla \phi|^3} - \frac{2\phi_x \phi_y \phi_{xy} + 2\phi_x \phi_z \phi_{xz} + 2\phi_y \phi_z \phi_{yz}}{|\nabla \phi|^3}. \quad (3.9)
\]

We sum this up in the following observation.

**Observation 3.1 (Smoothing term \( F_s = -\omega \kappa \))** Motion by curvature is known to have satisfactory geometric smoothing properties. When the coefficient \( \omega \) is a positive constant, points on the interface will move in the direction of concavity, i.e. in the direction of negative curvature, and the total variation of the interface will be reduced.

### 3.5.2 Balloon term

The balloon term, also called the expansion term, can informally be said to control “how hard the image is pulling on the isocontour”. It has its name from the idea that it “inflates” the interface like a balloon. The balloon term is a function of the image, and it often uses information about the image intensity within the object whose boundaries we are about to detect. This function should be high in areas where we are pretty sure there are no boundaries to be found, since the contour can move quickly.

---

\(^8\text{e.g. } \phi_x = \frac{\partial \phi}{\partial x} \text{ and } \phi_{xy} = \frac{\partial^2 \phi}{\partial x \partial y}\)
here. When the zero level set is close to a boundary, on the other hand, the speed caused by the balloon term should be small, so that the evolution of the interface slows down and the other two speed function terms (the smoothing term $F_s$ and the advection term $F_a$) should take over the decision of when to stop the deformation. Note that the balloon term is constant in time - it depends exclusively on the image intensity, and not on the level set function, as the two other speed function terms do.

In Slicer, the balloon term is defined as

$$c_I(\vec{x}) = \begin{cases} \exp\left(\frac{I(\vec{x}) - \mu}{\sigma^2}\right) - \tau, & \text{if } p_h = 0; \\ \exp\left(\frac{I(\vec{x}) - \mu}{\sigma^2}\right) - \tau, & \text{if } p_h \neq 0 \text{ and } I(\vec{x}) < p_h; \\ 1 - \tau, & \text{if } p_h \neq 0 \text{ and } I(\vec{x}) \geq p_h. \end{cases}$$

(3.10)

In the above equation $I = I(\vec{x})$ is the image as an intensity function, $\mu$ is the mean intensity and $\sigma$ is the standard deviation of the intensity of the object structure to segment. The parameter $\tau$ (called the probability threshold) is a threshold to allow a negative balloon term. We have a choice to set the probability high threshold, $p_h$, or leave it equal to zero. All the above terms will be explained below.

From the expression 3.10 we see that the balloon force will be strong in areas of the image with intensity close to the mean, while it will slow down the speed when moving away from the mean. If $p_h$ is defined (user optional), the balloon term is equal to the balloon term at the mean - i.e. the highest possible - for all intensity values higher than $p_h$. One choice is therefore to set $p_h$ equal to $\mu$ and set $\mu$ equal to the mean intensity of the part of the object that is close to its boundary, meaning that the intensity inside the object may be arbitrarily high (far above the mean) - the force

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3_4.png}
\caption{Examples of the balloon force as function of the image intensity. Left: $p_h = 0$. Right: $p_h = \text{mean intensity}$.}
\end{figure}
3.5 The speed function

Figure 3.5: The level set algorithm is run on the left image, which is the pre-processed and negated magnitude image. On the right, we see the segmentation result on top of the original image.

will still be strong in such areas. The motivation for such a specialized implementation is a wish to segment very bright objects (white matter in medical images). Figure 3.4 shows examples of the balloon term as a function of image intensity.

When using the level set tool in Slicer to segment other types of objects, one can ignore setting the probability high threshold. However, in a case such as ours, where we want to segment the dark areas of an image, we can exploit this feature by first negating all the intensity values in our images. Figure 3.5 shows an image, the negated image, and the output from the level set segmentation run on the negated imaged, but overlayed on the original.

The last term in equation 3.10 is $\tau$, which represents the probability threshold. The exponential term (the Gaussian function) in the equation can never become negative, nor can it become greater than one, which is the maximum value reached at the mean intensity, $I = \mu$. As mentioned $\tau$ is a threshold to allow negative balloon term, i.e. shrinking of the surface when the intensity is far from the intensity of the structure to segment. This means that the higher $\tau$ is, the more important it is to us that the speed function becomes negative, such that the forces acting on the interface are reversed when it has moved into an area with intensity values far from the mean. Here we assume that the interface is initialized inside the object. A common choice of $\tau$ is in the range $0.2 - 0.3$.

The observation below yields a summary of the properties of the balloon term.
Observation 3.2 (Balloon term $F_b = c_{I(\mathbf{x})}$) The balloon term is a function of the image intensity. Its purpose is to speed up the interface in areas where we know for sure that there is no edge in the image, by inducing contraction or expansion based on comparing image intensity with the intensity of the object to segment. In areas with high intensity variance, we suspect we are near the boundary, and therefore slow down the speed.

3.5.3 Advection term

Advection in general refers to the transport of some quantity through a vector field. The advection term in the level set segmentation problem uses information about the image to construct a vector field - it represents the outer energy working to deform the interface. This term is therefore said to control the stopping criteria of the interface evolution, i.e. detect when the zero levelset of $\phi$ has reached the boundary of the object of interest. The general form of this term is given as the (negative) inner product

$$-\nabla g \cdot \mathbf{n}$$

where $\mathbf{n} = \nabla \phi / |\nabla \phi|$ is the normal of the level set interface and $g = g(I)$ is a function of the image $I$. Remember that $\phi_t + F |\nabla \phi| = 0$ is equivalent to $\dot{x}_t = F \mathbf{n}$, thus the advection term contributes to moving points on the level set by $\dot{x}'(t) = \beta \mathbf{n}$ where

$$\beta = -\nabla g \cdot \mathbf{n}. \quad (3.11)$$

The function $g$ is constructed in such a way that it attracts the interface to the boundaries of the object, thus it is sometimes called an edge detector. Letting $g$ be a strictly decreasing function of the gradient of the image, it will take on small values where the image gradient is high, i.e. at the object boundaries. Suitable choices of $g$ include

$$g(I) = \frac{1}{1 + |\nabla \hat{I}|^p}, \quad (3.12)$$

where $\hat{I}$ is a smoothed version of the image $I$ and $p = 1$ or 2. When the interface is inside the object, and the gradient field of $g$ creates a negative inner product with the normal direction of $\phi$, then $\beta > 0$, and the interface will move in its outward normal direction towards the object boundaries. If the level set on the other hand crosses the boundary we are seeking, the

---

9$\hat{I}$ can be computed applying Gaussian filtering or some other image smoothing filter to $I$. 
Figure 3.6: The figure shows an image of a simple disk shaped object and a level set interface (in red) initialized inside it. The blue arrows along the curve represent outward normals to the level set, i.e. $\vec{n} = \nabla \phi / |\nabla \phi|$. The second figure shows a close-up look of the edge in the image. The gradient field $\nabla g$ of the edge detector function $g$ is overlayed on the image.

The gradient field of $g$ will coincide with the direction of $\nabla \phi$, yielding a positive inner product and $\beta < 0$, pulling points on the level set back towards the boundary. Thus the level set interface will eventually be forced to stay on the edges of the image.

Figure 3.6 shows a simple image $I$ together with the vector field of the edge detector function $g$ as given by equation 3.12 with $p = 2$ and where $\tilde{I}$ is $I$ convolved with a Gaussian of size $3 \times 3$ and standard deviation 0.5. An interface $\phi = 0$ and some gradient vectors of $\phi$ along the interface are also plotted. This example serves to illustrate what is pointed out in this section.

The purpose of the advection term may be summed up as follows.

**Observation 3.3 (Advection term $F_a = -\nabla g \cdot \vec{n}$)** The advection term’s task is to act as an edge detector. It adjusts the position of the interface when close to the real object boundaries.

### 3.5.4 Termination of the interface evolution

The level set interface is evolved according to the speed function $F$ until $F$ reduces to (near) zero and the interface stops. Since $F$ is composed of three terms, it is basically a problem of balance between the different speed terms. Note that the advection term $F_a$ and the balloon term $F_b$ tend
to shrink the contour (by becoming negative) if it crosses the object boundaries.

The evolution is driven by repeatedly updating the level set function $\phi$ for each elapsed time step $\Delta t$. Hence it is of course possible to stop the evolution after a predefined number of time steps, i.e. without the speed function vanishing. In our application we wish to locate the boundary of the frozen region. But since the frozen region extends into air, the boundary is not closed. We therefore have no reason to believe that the evolution of the level set interface will cease unless we limit the number of iterations. Figure 3.7 in the next section illustrates the evolving front at different time steps of the evolution.

### 3.6 Usage and results

As discussed earlier we perform level set segmentation on our images using existing software. The program we use is called Slicer and contains a module for level set segmentation. The program is developed for use in medical imaging applications, and it has some standard protocols that are specialized for such purposes. We selected the US (ultrasound) liver protocol within the level set module. Some of the parameters were adjusted to fit our images. This applies to for instance the choice of mean intensity, since the images are negated before given as input to the level set algorithm, cf. the discussion of the balloon term in section 3.5.2. The parameters are then as follows:

- **Initialization**: A sphere of radius 4 with manually selected center located approximately at the probe tip.
- **Dimension**: 3D.
- **Smoothing term**:
  - Smoothing type: Mean curvature.
  - Coefficient: 0.3
- **Balloon term**:
  - Mean intensity: $\mu = -1$
  - Standard deviation of intensity: $\sigma = 20$
  - Probability threshold: $\tau = 0.2$
  - Probability high threshold: $p_h = -1$ ($= \mu$)
  - Coefficient: 0.5
3.6 Usage and results

- **Advection term:**
  - Advection type: $g(I)$ as given by equation 3.12 with $p = 2$.  \(^{10}\)
  - Coefficient: 0.8

In addition to the above, there are parameters in order to control the size of the narrow band, the reinitialization frequency, the time step, and a few others. Adjustment of these parameters goes beyond the scope of this report, and we have relied on the standard implementation concerning such details.

The number of iterations (time steps) that were necessary was selected manually based on observation, and the evolution was stopped once the required shape was obtained. Between 200 and 250 iterations were adequate for our data sets. The average time spent per iteration was 0.051, 0.067, and 0.039 seconds for experiment 1, 2 and 3, respectively. It would generally not be a problem to let the interface evolve longer than the time that is strictly needed, thus we could have used 250 iterations for all the data sets if we were to avoid manual intervention. However, for practical reasons, as few iterations as possible is desirable, since it requires less computational cost. And more importantly, since the interface does not stop as it reaches air, it will just continue growing (until it reaches the border of the image domain). Thus the larger the region inside the interface, the larger the amount of data needed in order to represent it. The size of the interface is of importance to us considering how heavy the computations become later, as we are going to study geometric properties (curvature) and phase profiles of the level set interface.

In the illustrations in figure 3.7 we have chosen one slice from each of two different 3D image volumes, and show the result of segmentation within each slice at the selected time steps.

For the surface segmentation (see chapter 7) we have used Slicer to generate a triangulated surface from the final level set function, $\phi$. Slicer utilizes the Marching Cubes Algorithm in order to produce a triangulation that approximates the zero level set of $\phi$.

Users of Slicer have access to various functions that operate on the triangulated model. Slicer is built on the Visualization ToolKit (VTK), which is an open source software system for 3D computer graphics, image processing, and visualization. \(^{11}\) We will exploit the *decimation* and the *smooth-

\(^{10}\)With $g = \frac{1}{1 + |\nabla I|}$, the advection term $\nabla g \cdot \hat{n}$ can be written $g'(|\nabla I|) \left( H \cdot \frac{\nabla I}{|\nabla I|} \right) \cdot \hat{n}$, where $H$ is the Hessian of $I$. Slicer uses this formula, but *without* the term $g'(|\nabla I|)$, i.e. the advection term in Slicer is more correctly $(\nabla g \cdot \hat{n}) / g'(|\nabla I|)$.

\(^{11}\)VTK consists of a C++ class library, and several interpreted interface layers including
Figure 3.7: The level set interface is initialized as a small sphere approximately at the probe tip. The figure shows the result in one slice at different times $n\Delta t$ of the evolution, thus after one time step ($n = 1$) the interface can be seen as a circle shaped contour inside the frozen region. The interface evolves in the normal direction of the contour. In the upper image the contours corresponds to time steps $n = 1, 20, 50, 80, 110, 140, 170, 200, 230, 250$, and in the below image $n = 1, 20, 50, 80, 110, 140, 170, 200$. 
3.6 Usage and results

Figure 3.8: The left model shows the level set surface before smoothing. On the right, the surface mesh has been smoothed.

ingular functionalities that are implemented in Slicer via VTK. These operations allow us to improve the quality of our triangulated level set surface.

The smoothing operation adjusts point coordinates using a windowed sinc function interpolation kernel. A windowed sinc function is a standard signal processing low-pass filter. The effect is to make the mesh better shaped and the vertices more evenly distributed.

Decimation reduces the number of triangles in the mesh, i.e. removes vertices, as long as a good approximation to the original geometry is kept. Each vertex is classified and inserted into a priority queue. The priority is based on the error that will be caused if we delete the vertex and retiangulate the hole. Retiangulation is performed by edge collapse. Vertices that cannot be deleted or triangulated are not considered for deletion.

In Slicer the user may specify the number of decimate and smoothing iterations. The default is to perform 20 smoothing steps with a sinc filter and no decimation. We used 60 smoothing steps and 1 decimation step. Figure 3.8 shows a triangulated level set surface model before and after the application of smoothing.

\[
sinc(x) = \frac{\sin(\pi x)}{\pi x}.
\]
Chapter 4

Phase unwrapping

The MRI signal is a complex valued signal. The MR scanner’s reconstruction algorithm produces real and imaginary images, from which magnitude and phase maps can be created. These measures can be exploited in various ways to provide information about the imaged object. In this chapter we will explore how to interpret the phase of the signal, as well as a method called phase unwrapping. This method converts the original phase map to a more advantageous form of representation.

4.1 The phase of the complex MRI signal

The magnitude of the complex MRI signal is usually the only part which is used to produce contrast in MR images, such as in the image in figure 4.1. However, the phase contains vital information about the different types of tissue. The phase map gives information about the resonance frequency of the protons, [24]. The proton resonance frequency is closely related to tissue temperature. Our purpose is to provide input parameters to a temperature mapping algorithm. Thus it is of special interest to exploit the fact that the phase is very sensitive to changes in the tissue temperature.

A phase image is created from displaying the phase of the complex numbers (i.e. the angles) from the MRI scanner. The phase map thus consists of scalar values within the range from $-\pi$ to $\pi$. This map will therefore not distinguish between phase values which differ with a whole multiple of $2\pi$. We say the phase is wrapped, meaning its original value is replaced by its value modulo $2\pi$ whenever it initially falls outside the interval $[-\pi, \pi)$. The true phase rotation is therefore not known exactly. However, since we know that the wrapping procedure has only changed the original values of the image by adding or subtracting whole multiples.
of $2\pi$, we have a fair hope that it may be *unwrapped* without resorting to complicated computations. In the next section we will describe how a phase unwrapping algorithm was developed, based upon the assumption that the local phase map is continuous.

An example of a wrapped phase map may be seen in figure 4.2. This image is taken of an in-vitro bovine liver at room temperature, before any experiment was conducted. We expect the *true* phase to be varying continuously throughout the image slice, except in the parts of the image where there is no MRI signal. However, since the phase image in figure 4.2 represents *wrapped* phase values, it contains discontinuities along lines where the phase undergoes a jump of $2\pi$.

Figure 4.1 shows the corresponding image of the magnitude of the MRI signal, where the liver can be viewed against the signal void of the background, i.e. air and other surroundings. When the magnitude of a complex number is zero, the phase can be any random value, hence the black portions of the magnitude image will appear as noise in the corresponding phase image. Note that there are some small areas inside the liver that appear with little or no signal. These are mainly due to vessels which have been removed from the liver, small folds due to the placement on the table, and other imperfections or variations in the liver tissue.

To summarize, the signal void regions of the magnitude image explain
4.2 The general unwrapping procedure

Figure 4.2: MR image of bovine liver. Wrapped phase map.

the noisy regions of the phase image. The rest of the discontinuities in the phase image appear along distinct borderlines, called fringelines if we follow the terminology of Chavez et al. in [1]. These fringelines represent borders between adjacent pixels where phase wrapping seems to have occurred. The pixel value can be readily checked to be close to $-\pi$ on the dark side of such a border, while it makes a positive $2\pi$ jump as we move across to its much brighter neighbouring pixel, where the value is close to $\pi$.

4.2 The general unwrapping procedure

Let us denote the wrapped phase map by $\psi$, and use double subscripts whenever we want to access the value at a certain pixel index, e.g., $\psi_{i,j}$ would give us the value of the wrapped phase at the pixel in row $i$ and column $j$ of the image. In the same manner, let the unwrapped phase map be denoted by $\phi$.

The wrapped phase $\psi_{i,j}$ satisfies $\psi_{i,j} \in [-\pi, \pi)$ for all defined values of $i$ and $j$. Furthermore, we have the following relationship between the
38 Phase unwrapping

wrapped (ψ) and unwrapped (φ) phase maps

\[ \psi_{i,j} = \phi_{i,j} + 2\pi n_{i,j}, \]  

(4.1)

where \( n_{i,j} \) is the positive or negative integer such that \( \psi_{i,j} \) lies within the required \([-\pi, \pi)\) range. In order to make the notation easier, we will define a \textit{wrapping} operator \( W : \mathbb{R} \rightarrow [-\pi, \pi) \) by

\[ W(x) = x + 2\pi n \quad \text{with} \quad n \in \mathbb{Z} \quad \text{such that} \quad W(x) \in [-\pi, \pi). \]

In terms of this operator, the relationship between the unwrapped and wrapped phases becomes \( \psi = W(\phi) \).

Assuming the image dimensions are \( m \times n \) we may think of \( \{\psi_{i,j}\}_{i,j=1}^{m,n} \) and \( \{\phi_{i,j}\}_{i,j=1}^{m,n} \) as discrete function samples from their respective underlying continuous functions, \( \psi(x, y) \) and \( \phi(x, y) \), where the samples are taken at grid points \((x_i, y_j)\), each at the midpoint of the pixel with the corresponding indices. Furthermore, we may then think of the phase difference between neighbouring pixels as an approximation to the partial derivative in the specified direction. For instance, let \( \Delta \psi^1 \) define the phase difference of \( \psi \) in the direction of increasing first variable, as indicated by the superscript. At the \((i, j)\)’th pixel center we have

\[ \frac{\partial \psi}{\partial x}(x_i, y_j) \approx \Delta \psi^1_{i,j} = \psi_{i+1,j} - \psi_{i,j}, \]  

(4.2)

defined for all indices \( i \) and \( j \) within the range of the underlying image, for which the expression is valid.

The idea is now that if we somehow knew the gradient of the unwrapped (true) phase map, we could simply find the map itself by taking an integral sum of this gradient along some path in the image. It can be shown that under certain assumptions on \( \phi \), we already have all the information available; simple arithmetic operations applied to the wrapped input phase would yield such a true phase gradient map.

In order to show how this works, let us consider an example where we use the first component of the gradient, as outlined above. Equations 4.1 and 4.2 yield

\[ \Delta \psi^1_{i,j} = \psi_{i+1,j} - \psi_{i,j} \]

\[ = (\phi_{i+1,j} + 2\pi n_{i+1,j}) - (\phi_{i,j} + 2\pi n_{i,j}) \]

\[ = (\phi_{i+1,j} - \phi_{i,j}) + 2\pi (n_{i+1,j} - n_{i,j}) \]

\[ = \Delta \phi^1_{i,j} + 2\pi k, \]
4.2 The general unwrapping procedure

where \( k \) is taken to be the integer \( k = n_{i+1,j} - n_{i,j} \). Applying the wrapping operator to the result, we see that

\[
W(\Delta \psi_{i,j}^1) = W(\Delta \phi_{i,j}^1 + 2\pi k) = \Delta \phi_{i,j}^1 \quad \text{if} \quad \Delta \phi_{i,j}^1 \in [-\pi, \pi) .
\]

Note that the above equality holds only in the case that \( k = 0 \), meaning \( n_{i+1,j} = n_{i,j} \), i.e. the two neighbouring pixel phase values have been wrapped by adding the same multiple of \( 2\pi \). This tells us that as long as the constraint \( \Delta \phi_{i,j}^1 \in [-\pi, \pi) \) holds, the true phase map derivatives are known and equal to the corresponding wrapped derivatives.

Established unwrapping methods usually take advantage of this assumption, and treat the exceptions in various ways. In [1] it is differentiated between two types of borderlines in phase images; fringelines and cutlines. While fringelines are borders due to wrapping of the phase, cutlines appear where the true phase has undergone a rotation greater than \( \pi \) (in absolute value) from one pixel to the next, thus the difference between adjacent pixel values are not within the desired \([-\pi, \pi)\) range.

When no cutline is present, it is straightforward to determine the exact value of the phase at any point by adding or subtracting a multiple of \( 2\pi \) each time a fringeline is crossed, [1, p.966].

The quoted sentence sums up the approach which we have attempted to take in our work, and will be thoroughly described in the following section.

According to [1] we can divide present phase unwrapping solutions into two main categories: 1) path-following methods, and 2) minimum normalization methods. The second category will not be further discussed here, but the idea is based on the requirement that the phase differences of the unwrapped image should agree with those of the wrapped phase in the minimum \( L^p \)-norm sense. The first category develops the idea discussed above of integrating the phase gradient map along some path. Since the gradient can only be known modulo \( 2\pi \) from the wrapped phase, it is important that the \( \Delta \phi \in [-\pi, \pi) \) condition is met, as integrating across pixels where this does not hold will cause an error which later will propagate across the rest of the final unwrapped image. The path-following methods must therefore focus on avoiding any error propagation. This may be achieved through guiding the path of integration so that it avoids using any true phase differences which are outside the range \([-\pi, \pi)\).

In this paper we have chosen a somewhat simplified path-following approach, as we do not need a general phase unwrapping algorithm, but
one that works well on the MR images we will be working on in the context of hepatic cryo surgery, and the problem of separating frozen liver from air. The phase unwrapping algorithm developed here will serve as a preprocessing step before we can study the real phase, with the goal of detecting discontinuities in phase due to temperature changes. The discontinuous wrapped phase would clearly not be applicable for this task. The interest of keeping the algorithm simple and stable has been stressed at each step of the development.

4.3 Our phase unwrapping method

In the current section we will present our solution to the phase unwrapping problem. The main steps of the algorithm will be outlined before we elaborate on each of them. We then demonstrate the use of the method displaying some example images along with a discussion. Pseudocode for the unwrapping routine is provided at the back of the chapter.

4.3.1 Algorithm outline

The basic idea of our method is to traverse all the pixels in the image, by letting a region grow from a *seed* pixel, searching to detect *fringelines* (cf. section 4.2) whenever the growing region encounters them. Each time such a fringeline is crossed, the adequate multiple of $2\pi$ must be added, in order to reverse the phase wrapping.

Since we only need to find the true phase up to an additive constant (i.e. a constant that is the same for the whole image), the starting point of the algorithm may in theory be any pixel in the image. We must however make sure that the seed pixel is within the imaged object (we cannot choose to start at a pixel in the region with no signal), and secondly we must ensure that we do not choose a value that is corrupted due to noise. The last requirement will be fulfilled by selecting the seed in a neighbourhood of pixels where the signal is (relatively) high. The unwrapping procedure is based on finding the unwrapped phase of a pixel from (one or more of) its neighbours $^1$, assuming their unwrapped phase values are already known. The algorithm will therefore gradually expand the region of pixels.

The output of the algorithm will be an *unwrapped phase image*, in which all the pixels where an MRI signal is present have been set to what is be-

$^1$We consider neighbourhoods in regular 8-connectivity.
lieved to be their true phase value. The effect of this is of course that those parts of the image where there is no signal (or the signal is very low) will not be changed. In order to determine how strong the signal is, we will use the magnitude image. Thus input will be a set of two images consisting of the magnitude image and its corresponding wrapped phase image.

As an alternative to building up a region of pixels by changing values each time a fringeline is crossed, we could instead have utilized the gradient map and integrated the whole image. However, this would severely complicate the algorithm; each pixel would need information about what part of the path it belongs to, and we would have to keep track of the phase changes made along the preceding part of that path. Our simplified version avoids such complications. It is also worth noting that the gradient is believed to be approximately constant except at the fringelines. This suggests that in either case a differentiation (i.e. a gradient map calculation) followed by an integration may cause superfluous work.

### 4.3.2 Region growing

As mentioned in the previous section, life would be easier if we could assume that the true phase never made a jump in values outside the range of \([-\pi, \pi)\) between contiguous pixels. Since noise will virtually always be part of the image, this is unlikely to be the case. The signal may be higher in some areas than in others. Trusting the signal in areas with a high amount of noise, might cause errors, which later propagate along the path of traversal.

Since we cannot rely on pixels with noisy values, as this may contradict our assumption that all adjacent pixel differences are within the unambiguous range of \([-\pi, \pi)\), the approach is to postpone dealing with those pixels as long as possible.

#### Thresholding

The algorithm starts by locating the pixel with the highest average neighbourhood signal intensity. This pixel will be used as seed pixel for the growing region. The average neighbourhood signal intensity at pixel \((i, j)\) may be denoted by \(\mu_{i,j}\) and approximated by computing the mean intensity in a \(k \times k\) pixel neighbourhood centered at \((i, j)\), with \(k\) a positive, odd integer.

---

2True phase value meaning here the correct value up to an additive constant per image.
In 256 × 256 pixel images as the ones we are operating on, a suitable choice is e.g. \( k = 9 \).

Once we have decided where to start the traversal of the image, the unwrapped phase is initialized by setting its value at the seed pixel equal to the wrapped phase value at the corresponding location. As soon as a pixel has obtained its unwrapped phase value this pixel will be marked as “set”.

Region growing will originate from the seed and new pixels will be added into the final image as they become immediate neighbours of the growing area of marked pixels. Before a pixel is included, we must find its true phase value. (For details on how this value is found, see the next section).

The order in which we choose to visit the pixels must be considered carefully, as it can be thought of as the “path of integration” (cf. section 4.2). Unstable values must be left out from the calculations as far as possible, or not be trusted at all, or otherwise they will influence on the result in a negative manner. This is why instead of including all pixels hit by the region as it grows, we only allow those pixels where the magnitude of the MRI signal is above a certain (strict) threshold \( T_1 \). Region growing proceeds until there are no more neighbours where the signal is strong enough to pass the threshold \( T_1 \). The algorithm then chooses a lower threshold \( T_2 \) based on the assumption that (almost) all pixels belonging to the foreground of the image (i.e. belonging to the liver) should be included. Then region growing can continue in the same manner as before. In a last step we lower the threshold even further, to an intensity level \( T_3 \), where even pixels with only a relatively low signal, will be included. Pixels passing the third threshold will often represent signal noise in the image, and it is therefore important to have as much information as possible about the phase values in a neighbourhood around such a pixel before we compute its true phase value. This explains why we choose to gradually advance the region growing through three steps. Only the pixels where we have “certain” information (i.e. where the magnitude is greater than \( T_1 \)) are treated at first, hopefully yielding enough information to correctly set the pixels passing the second threshold, and in the end, filling in the remaining areas of foreground by allowing also the pixels where the signal is low to become unwrapped. As can be seen in the example images, the liver usually contains some smaller areas of signal void, due to arteries etc. The phase will naturally make jumps here, and pixels where there is no signal, will not be unwrapped, even when they are surrounded by pixels representing the foreground.

The thresholds are set by first estimating the average tissue intensity,
4.3 Our phase unwrapping method

Figure 4.3: The magnitude image left gives a measure on how strong the intensity is in different parts of the image. The colors in the right figure indicate the strictest threshold each pixel passes. Pixels with intensity higher than T1 are red, higher than T2 blue, and T3 green.

The unwrapping thresholds $T_1$, $T_2$ and $T_3$ are then set to 100%, 60% and 20% of the average tissue intensity respectively, see figure 4.3.

computed as the average intensity of all pixels with signal stronger than a temporary threshold, $t_{temp}$. The threshold $t_{temp}$ is defined as the level of the histogram where the cumulative sum of pixels in the lower levels equals $x$ percent of the total number of pixels in the image, where $x$ is a parameter given by the user. Thus $x$ must be adjusted for each image series and should reflect the approximate fraction of background (i.e. non-object) pixels. (We used an estimate of 82% background in our images).

One may be aware that some of the pixels passing the third intensity threshold may actually represent noise in the background pixels, and lie outside the portion of the image corresponding to the liver. However, these pixels will never be treated during the unwrapping algorithm, since they will not be encountered as neighbours of the growing area.

Note that there may exist more than one area that requires unwrapping per image. When the object foreground consists of two (or more) disconnected areas (e.g. when the unfrozen liver consists of two parts) each part must be unwrapped separately. (An example of such an image will be presented together with the results below). In practice such occurrences are detected as follows. When the unwrapping ceases (i.e. no more pixels neighbouring the region pass any of the thresholds) then we search the remaining image (i.e. the “unset” pixels) for pixels with high intensity value. If there are several such pixels we have reason to believe that there is a part of the image yet to explore, and we initialize the unwrapping process once
more with a seed from the unprocessed area. Note that if this happens, the additive integration constant *may* end up being different for each of the separately unwrapped parts of the image.

### 4.3.3 Fringeline detection

While the region grows, the encountered pixels will be compared to their previously visited pixel neighbours. As long as the values along the path changes continuously, nothing will be done, i.e. values from the wrapped phase map will be copied directly to output.

However, when we reach a fringeline, we must add the correct multiple of $2\pi$. The question is how to detect that the growing region of pixels has encountered a fringeline. Since we assume some degree of continuity, we simply check whether the difference between the pixel and its neighbours is below a certain set threshold. More specifically we use the median of those neighbours whose value has already been determined when measuring the difference. When the difference is *not* below the threshold, we assume that the pixel belongs to a fringeline, and thus a (positive or negative) multiple of $2\pi$ must be added. We do not want to be exceedingly strict in our continuity assumption, as there are several reasons to believe that it will not always hold locally. Since the threshold is meant to alert us whether a pixel belongs to a fringeline or not, it must be able to detect phase jumps around $2\pi$ in absolute value. For this reason, a suitable threshold value is $2\pi - \epsilon$, where $\epsilon$ is some small tolerance.
Whenever the absolute value of the phase difference between adjacent pixels exceeds the threshold, we consider it as belonging to a fringeline. Every time this occurs, the correct multiple of $2\pi$ has to be computed from the difference and added. Thus, the first time a fringeline is discovered, a single multiple will be added to the pixel on the fringeline, and this addition will disseminate to the following pixels, until a new fringeline is reached, where the single multiple will no longer yield a phase difference within the tolerance, and consequently the addition of a second multiple will be called for, and so on.

Figure 4.5: MR image acquired during studies of cryo ablation in living pigs.

4.4 Results and discussion

The final result of the example used in figure 4.3 is shown in figure 4.4. Figure 4.7 shows how unwrapping worked on example slices from other cryo experiments. (Some additional examples of unwrapped phase images may be seen in chapter 5, section 5.4, where segmentation is performed using these images).

In clinical situations the MR images will contain other organs, skin, fat etc. in addition to the liver. An example of such an image is displayed in figure 4.5. Since our unwrapping algorithm is developed based on images containing an isolated liver detached from the body, it may have problems if dealing with more complex phase maps. A solution might be to seclude the liver by segmentation (methods for doing this already exist, see e.g. [7]), and apply the unwrapping routine only to the part of the image that is recognized as belonging to the liver.
Figure 4.6 shows an example where the unwrapping algorithm fails to provide a perfect result. The magnitude image shows that the intensity of the signal was relatively low in this case, which may have affected the unwrapping process. A solution may be to adjust the input parameter used by the unwrapping algorithm, i.e. the fraction of background pixels. In order to limit the user interaction as much as possible we provided a parameter for each image volume, instead of each image slice. It may be hard to optimize the choice of parameter when the variation in signal strength between images from the same volume is as large as it was here. However, when the signal is good, there should be no problem. We obtained a satisfying result in all the images where the signal intensity was normal, i.e. as expected, or better than normal.

We performed some tests of the algorithm using fewer (i.e. one or two) threshold values, as well as tests with a larger number of thresholds. With one or two thresholds the output of the algorithm was not satisfactory, presumably because it trusted some corrupted values at a too early stage of the unwrapping process. On the other hand, augmenting the number to four thresholds did not seem to improve the algorithm. The thresholds must be computed at the beginning of the unwrapping process, and a higher number of thresholds causes more “delay” during the traversal. In other words; we want to use as few thresholds as possible, while at the same time sufficiently many in order to produce the correct output. We therefore conclude that using three thresholds should be adequate in situations similar to ours.

Pseudocode is provided starting on page 48.

Figure 4.6: A magnitude image where the signal intensity is relatively low (left) and the result of phase unwrapping (right).
Figure 4.7: From above: Magnitude, wrapped and unwrapped phase MR images of the same slice. The right and left examples are from two different experiments.
Algorithm 1 Phase unwrapping

1: function UNWRAP(mImg, wpImg, bf) \(\triangleright\) mImg: magnitude image, wpImg: wrapped phase image, bf: approximate fraction of background pixels in the magnitude image

2: upImg ← phImg \(\triangleright\) Initialize upImg: unwrapped phase image

3: slmg ← zeros(size(magnImg)) \(\triangleright\) Initialize all pixels as not set (i.e. as zero) in slmg

4: mvalImg ← meanValueFilter(mImg)

5: thresholds ← SET-THRESHOLDS(bf) \(\triangleright\) See algorithm 2

6: repeat

7: Make empty container: unwrappedArea \(\triangleright\) unwrappedArea will successively include pixels as they obtain unwrapped values

8: seed ← pixel with maximum mean value among pixels corresponding to slmg = 0

9: Include seed in unwrappedArea

10: for \(i \leftarrow 1, 3\) do

11: threshold ← thresholds(i)

12: repeat

13: Make empty container: growArea \(\triangleright\) growArea will successively include pixels that become neighbours of the unwrappedArea AND pass the current threshold

14: for \(p \leftarrow \text{unwrappedArea(firstIndex), unwrappedArea(lastIndex)}\) do

15: for \(n \leftarrow \text{first neighbour of } p, \text{ last neighbour of } p\) do

16: if slmg(n) = 0 AND mImg(n) > threshold then

17: Include n in growArea

18: end if

19: end for

20: end for

21: for \(q \leftarrow \text{growArea(firstIndex), growArea(lastIndex)}\) do

22: if slmg(q) = 0 then

23: SET(q) \(\triangleright\) See algorithm 2

24: slmg(q) ← 1

25: Include q in unwrappedArea

26: end if

27: end for

28: until growArea is empty \(\triangleright\) growArea will be empty when no more neighbours pass the current threshold

29: until \((mImg(p) > \text{threshold}(1) \text{ AND } slmg(p) = 0)\) holds for < 50 pixels \(p\) \(\triangleright\) i.e. until the unwrapping has finished

30: end function
Algorithm 2 Phase unwrapping help functions

1: \textbf{function} SET-THRESHOLDS($bf$) \\
2: \hspace{1em} $mMax \leftarrow \max(mImg)$ \\
3: \hspace{1em} $hist \leftarrow \text{histogram of } mImg$ \\
4: \hspace{1em} $cumSum \leftarrow \text{cumulative sum of } hist$ \\
5: \hspace{1em} $bgp \leftarrow bf \times xdim(mImg) \times ydim(mImg)$ \Comment{$bgp = \text{estimated number of background pixels}$} \\
6: \hspace{1em} $tTemp \leftarrow \text{level of } hist \text{ where } cumSum \geq bgp$ \\
7: \hspace{1em} $\mu \leftarrow \text{mean intensity of } mImg(mImg > tTemp)$ \\
8: \hspace{1em} $\text{thresholds}(1) \leftarrow \text{floor}(\mu)$ \\
9: \hspace{1em} $\text{thresholds}(2) \leftarrow \text{floor}(0.6\mu)$ \\
10: \hspace{1em} $\text{thresholds}(3) \leftarrow \text{floor}(0.2\mu)$ \\
11: \textbf{end function}

1: \textbf{function} SET($p$) \Comment{Set the unwrapped value of pixel $p$} \\
2: \hspace{1em} \text{Make empty container: } neighbours \\
3: \hspace{1em} \textbf{for} $n \leftarrow \text{each neighbour of } p$ \Comment{Pixel neighbours are considered in a regular 8-connected neighbourhood} \\
4: \hspace{2em} \textbf{if } sImg(np) = 1 \textbf{ then} \\
5: \hspace{3em} \text{Include } mImg(np) \text{ in } neighbours \\
6: \hspace{1em} \textbf{end if} \\
7: \hspace{1em} \textbf{end for} \\
8: \hspace{1em} val \leftarrow \text{median(neighbours)} \\
9: \hspace{1em} diff \leftarrow upImg(p) - val \\
10: \hspace{1em} n \leftarrow \text{floor}(\frac{|diff|}{2\pi}) \\
11: \hspace{1em} \textbf{if } \text{diff} > 0 \textbf{ then} \\
12: \hspace{2em} upImg(p) \leftarrow upImg(p) - 2\pi n \\
13: \hspace{1em} \textbf{else} \\
14: \hspace{2em} upImg(p) \leftarrow upImg(p) + 2\pi n \\
15: \hspace{1em} \textbf{end if} \\
16: \hspace{1em} lim \leftarrow 2\pi - \varepsilon \Comment{cf. section 4.3.3} \\
17: \hspace{1em} diff \leftarrow upImg(p) - val \\
18: \hspace{1em} \textbf{if } |diff| > lim \textbf{ then} \\
19: \hspace{2em} \textbf{if } \text{diff} > 0 \textbf{ then} \\
20: \hspace{3em} upImg(p) \leftarrow upImg(p) - 2\pi \\
21: \hspace{2em} \textbf{else} \\
22: \hspace{3em} upImg(p) \leftarrow upImg(p) + 2\pi \\
23: \hspace{2em} \textbf{end if} \\
24: \hspace{1em} \textbf{end if} \\
25: \textbf{end function}
Chapter 5

Segmentation based on a phase strategy

One advantage of using MRI as imaging tool when performing cryo ablation is that it has certain properties that enable 3D temperature mapping during treatment, [24]. As mentioned in section 1.2, one way to produce a 3D temperature map in the frozen region is to solve a simplified bio-heat differential equation. The method is based on using MR images to obtain information about the location of the border of the frozen region. The 0°C isosurface is found by segmentation of the images. Previously the segmentation has been done manually, while in this thesis we approach the same task trying to develop an automatic segmentation algorithm.

We divided the segmentation process into two main steps, cf. section 1.2.1. The second step of the segmentation is to separate frozen tissue from air where the ice-ball extends to the surface of the liver. In order to find this transition front our hope was that the temperature sensitive phase maps from MRI scanning of the liver could be exploited. Unfortunately, this approach has not shown promising results. However we would like to discuss our hypothesis in the current chapter, and argue why it seems to fail.

5.1 PRF temperature mapping

One technique to perform temperature mapping using MRI is to exploit the fact that when the tissue temperature changes, the temperature difference is proportional to the phase difference, where by phase we mean the phase of the complex valued MRI signal. Denoting the phase difference by $\Delta \phi$ and the temperature difference by $\Delta T$, this relationship may
be expressed

\[ \Delta \phi = 2\pi \gamma B_0 \alpha T_E \Delta T, \]

([18]), where \( \gamma \) is the gyromagnetic ratio of hydrogen, \( B_0 \) is the main magnetic field strength, \( T_E \) is the echo time \(^1\) in seconds, and \( \alpha \) is the temperature change of the chemical shift in ppm/°C. \(^2\) The value of \( \alpha \) has been calculated from experiments where the temperature was measured using another instrument than MRI. It has been shown that this coefficient is independent of tissue type, and that its value is \(-0.0098 \pm 0.0005\) ppm/°C, cf. [24] and references therein.

Provided the initial temperature is known, a temperature map may be calculated from equation 5.1. The phase difference \( \Delta \phi \) is found by subtracting a current phase image from a baseline phase image, i.e. an image acquired before thermal therapy has been conducted. This technique is known as Proton Resonance Frequency (PRF) temperature mapping. Some adjustments should be made to correct for movement of tissue during the treatment. MRI thermometry methods are also very sensitive to the phase drift induced by the instability of the scanner, causing the spatially local magnetic field that the proton is experiencing to vary temporarily. Under the condition that the temperature is kept constant in time, only the phase drift can contribute to phase changes in the MR image, [6]. Such corrections may be hard to perform. However, since the temperature will change in the neighbourhood of the ice-ball under cryo therapy, we want to study phase sensitive MR images in order to find out whether they can provide useful information about the border of the frozen region.

As mentioned in section 1.1.1 in chapter 1, frozen tissue is signal void when using conventional MRI pulse-sequences. Nevertheless, specialized pulse-sequences with very short echo time (TE) can actually produce some signal from frozen tissue, cf. section 2.1. Thus it is possible to perform PRF temperature mapping during a freezing process, as long as there is a signal. However, it has not been established whether this has a linear dependence on temperature.

### 5.2 Clipping the level set contour

We want to avoid using a baseline image, since movement of the imaged organs caused by breathing and/or surgical operation complicates such an

---

1The echo time TE is the time between the start of the applied RF pulse and the maximum in the signal sent back from the tissue. This parameter is used by the MRI scanner.

2ppm = parts per million.
5.2 Clipping the level set contour

Figure 5.1: Two examples using level set segmentation in order to find the frozen region. The level set interface is shown in red. In the left image the entire ice-ball is contained inside the liver, while in the right the ice-ball extends to the liver surface.

approach. The goal is therefore to perform direct segmentation on images acquired at a given current stage of cryo therapy only, without depending on reference images.

As outlined in the introduction to the segmentation process (section 1.2.1) we want to use level set methods to segment the border between frozen and non-frozen tissue. Since there is no border between frozen tissue and air in MR images, the level set interface will grow out of the frozen region in places where it extends to the surface of the liver. Figure 5.1 shows two examples. (Although these are 2D image slices, the method was run on an image volume (a set of continuous slice images) in three dimensions. This way the product is a level set surface, hence the contours in the images are the intersections of this surface with the respective slices.) In the left image the ice-ball is entirely inside the liver, while on the right we see what happens when the contour reaches the liver surface. In such cases we need to perform a second segmentation step in order to clip the contour at the border.

We will study more closely the cases in which the frozen region has a border against air so that the level set interface inflates out of the liver parenchyma and thus needs to be clipped. Because of the invisible border between air and frozen tissue, the boundary cannot be extracted directly from the magnitude images. We will instead try to infer where the boundary should lie, based on the level set interface.

The shape of the level set interface above the liver depends on what objects might reside close to the surface. According to Heuch et al. ([8]) the interface will either wrap around the corners of the unfrozen tissue and
develop a characteristic “mushroom shape”, or it will continue straight up. The first occurs when there is empty space above the liver, while the latter shape is presumably developed when other objects (i.e. fat, skin or other organs) reside close to the liver surface at the insertion point of the probe(s), and the arrangement of objects outside the liver prevents the contour from bending at the surface. We will focus on studying the so-called mushroom-shaped interfaces, as our experiments were conducted using separated in-vitro livers and therefore with no other objects involved in the images.

5.3 Phase profiles

One approach is to analyze phase profiles along the edges as defined by the level set method, and establish criteria to determine the border between tissue and air. We expect the phase to roughly vary constantly (cf. chapter 4) along the part of the interface that defines the edge between frozen and non-frozen tissue, since the temperature is constant here. At the surface however, where the interface bends away from the frozen region and follows the outer face of the liver, the temperature will change, hence one would expect also the phase to change significantly, and possibly even experience a discontinuity.

When applying the phase strategy, the clipping of the interface will be done per slice by locating points on the contour and connecting them by line segments. The parts of the interface that reside inside the liver will be kept together with the new line segments, while the parts that are outside the liver will be discarded, so that the resulting new contour estimates the boundary of the frozen region. The points should thus represent intersections between frozen tissue and air. As the contour wraps around the corners of the frozen area, it will follow the liver surface, and we expect changes in phase values as we move away from the frozen area. In order to study this more closely we will plot a phase profile along the contour. This involves imposing the level set interface extracted from the magnitude image upon the corresponding unwrapped phase image. The level set contour is represented by a discrete connected set of points in the image coordinate system. The profile will be a function of this series of points, and the function values will be the phase values extracted from the image at the locations where the image is hit by the level set contour.

More specifically, level set segmentation produces an image with pixel values corresponding to the height of the level set function, cf. chapter 3. The level set interface is found as the zero level isocontour of the level
set function. We have implemented the Marching Squares Algorithm for the purpose of extracting the contour. This method operates on a 2D grid and draws lines along linearly interpolated values along the edges of the squares in the grid. The grid points, i.e. the corners of the squares, are in our case defined to lie at the image pixel centers. The resulting level set contour therefore consists of an ordered set of vertices whose coordinates lie on the gridlines, i.e. on a sub-pixel level.

Formally we will follow the notation from an article on a discrete dynamic contour model (snake) by Lobregt and Viergever, see [15], and also used by Heuch et al. in [8]. We denote each vertex by \( V_i \), so the contour consists of connecting the \( n \) vertices \( \{V_i\}_{i=0}^{n-1} \) by straight line segments. Our task is to study how the phase behaves as we move along the contour, thus phase values are found at each vertex of the interface.

Phase values are taken from the phase MR images that were acquired during PRF scanning (cf. section 2.2), and later unwrapped by our algorithm as described in chapter 4. We do not want to rely on just the single pixel underneath each vertex of the level set contour, so we find a vector of length \( m \) units (one unit in the image coordinate system equals the length of one pixel edge) taken in the negative normal direction from

\[ \begin{array}{c}
\text{Figure 5.2: The arrow indicate the negative normal direction from the vertex marked on the contour. The vertex and three additional points found in this direction determines which pixels we use to estimate the phase value at the location. These pixels are shadowed.}
\end{array} \]

\[ \text{In [15] the normal direction is called the radial direction. Hence the unit vector in this} \]
the vertex on the interface. Here \( m \) is a small positive integer. We have chosen to include 3 phase values in addition to the one at the vertex itself, thus using \( m = 4 \). Each point of the phase profile will thus be the average of the phase values at the pixels underneath those \( m \) points, see figure 5.2. An exception must be done in cases where the contour hits points that reside outside the area that has become phase unwrapped, hence the phase value at the vertex is unknown. This may occur (although rarely) in places where the level set contour has not succeeded to completely reach onto the border of the frozen region. We overcame this problem by letting the phase value at the vertex in question be estimated by the remaining \( m - 1 \) points in the normal direction, provided their unwrapped phase values were known.

Hence the profile is a function \( F = F(i) \) defined for \( i = 0, 1, \ldots, n - 1 \) where \( F(i) \) is the estimated phase value at vertex \( V_i \). The profile should also be smoothed in the \( x \) direction to correct for outlier values (noise). One choice is to smooth the phase profile \( F \) by defining a new function \( G \) where each component is a weighted average of the values at neighbouring vertices, e.g.

\[
G(i) = \frac{1}{16} F(i - 2) + \frac{1}{4} F(i - 1) + \frac{3}{8} F(i) + \frac{1}{4} F(i + 1) + \frac{1}{16} F(i + 2),
\]

which is defined at all vertices \( V_i, i = 0, \ldots, n - 1 \), if we let \( V_{-1} = V_{n-1}, V_{-2} = V_{n-2}, V_n = V_0 \) and \( V_{n+1} = V_1 \), as a consequence of vertex \( V_0 \) and vertex \( V_{n-1} \) being connected.

The normal direction is derived from the coordinates of the boundary vertices. We first find the tangential direction at an interface vertex \( V_i \) by defining a locally tangential unit vector \( \hat{t}_i \) as

\[
\hat{t}_i = \frac{\hat{d}_i + \hat{d}_{i-1}}{||\hat{d}_i + \hat{d}_{i-1}||}
\]

where \( \hat{d}_i \) is the unit vector pointing in the direction of the edge segment leaving from vertex \( V_i \). Thus if the position of vertex \( V_i \) is given by a vector \( p_i \) then

\[
\hat{d}_i = \frac{d_i}{||d_i||} \quad \text{where} \quad d_i = p_{i+1} - p_i.
\]

Note that the vertices \( V_i \) for \( i = 0, \ldots, n - 1 \) should be oriented in clockwise manner along the boundary enclosing the frozen region. Now we

---

\(^4\)The weights are the normalized binomial coefficients \( \binom{n}{i} / \sum_{k=0}^{n} \binom{n}{k} \) for \( i = 0, \ldots, n \), with \( n = 4 \).
5.3 Phase profiles

Figure 5.3: The local tangential direction at $V_i$ is defined as the normalized sum of the normal vectors of the two edges meeting at the vertex. Remember that the position of $V_i$ is represented by the vector $p_i$. The local normal direction is found by rotating the tangent by 90°. (The figure is redrawn from [15]).

can define the unit normal direction vector $\hat{r}_i$ by rotating the unit tangent vector by 90°, i.e.

$$\hat{r}_i = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \hat{t}_i.$$  

See figure 5.3 for an illustration.

It makes little sense to include phase values that are not unwrapped (cf. chapter 4) in the phase profiles. We therefore chose to exclude vertices where no unwrapped values were found at the $m$ points in the normal direction. The phase profile $F = F(i)$ will consequently be plotted for a subset $\{V_i\}_{i=p}^{q} \subset \{V_i\}_{i=0}^{n-1}$ of all the vertices. Hence the part of the phase profile corresponding to the part of the contour which resides outside the liver parenchyma will be discarded. In some cases the liver is split in two parts as a result of the frozen region extending all the way through the liver and to the opposite side of the probes. The phase profile in such occurrences is then divided in two separate plots, each corresponding to a subset of contiguous vertices taken from the level set interface, see figures 5.4 and 5.5 for examples.
Notice that the model is closed, so the first vertex $V_0$ and the last vertex $V_{n-1}$ are connected. An open model would also be possible, but it would require special measures for the calculation of the local tangential and normal vectors at the two end vertices $V_0$ and $V_{n-1}$, since an open model implies that they are not connected and therefore have only one neighbour each.

### 5.4 Results and discussion

In this section we will present examples of phase profile plots from PRF images acquired during experiments. The profiles are taken along selected parts of the level set contours as presented on the images in figures 5.4 and 5.5. Note that the level set contours are extracted from the magnitude images, while the values in the plots are captured from the corresponding unwrapped phase images. In the figures we show the contours overlayed on the unwrapped phase. Hence we are hoping that the phase signal may bring forth the deficient information in order to perform the segmentation of the frozen region in the corresponding magnitude image.

Studying the plots we notice a tendency in the phase to drop at the part of the contour lying at the border of the frozen region. Be aware of the fact that although there are parts of the contour where the temperature is expected to be approximately constant, the phase will not be constant here, but rather continuously varying, as we must account for phase drift in the image. As mentioned in section 5.1, the phase drift is a result of the image volume experiencing slightly varying magnetic field strength due to placement of the coils inside the MRI scanner.

We have manually selected points on the phase profile, and indicated them with number tags on the plot. In the image parallel to the plot we have then marked and numbered the vertices belonging to each selected point accordingly.

In some instances the points where the contour should be clipped were fairly easy to identify from studying the phase profile alone. As already mentioned, the profile usually descends slowly in the beginning, and then makes a sudden change and drops more drastically, in some cases after a small increase. Thus the first clip point will lie on a local maximum. After reaching its global minimum point about half way into the frozen area, the behaviour of the profile is mirrored onto the last half of the plot. The selected points are usually at a point where the profile starts its main decrease, or (on the second half), where it ends its main increase. Note however that this description fails even in just the few cases presented in
5.4 Results and discussion

Figure 5.4: On the left the image is shown with a blue and red level set contour. The phase profile is plotted for the subset of vertices constituting the red part of the contour. The vertices are numbered clockwise. The red labeled points in the image correspond to the points marked at the phase profile which is plotted on the right. The phase values in the plot are averages of 4 points taken in the normal direction from (and including) the vertex. The plots that are divided in two parts, each corresponds to one of the two selected (red) parts of the level set interface, as can be seen in the accompanying image.
Figure 5.5: See figure 5.4 for an explanation.
the figures of this chapter. Observe for instance the second and seventh clip points in the bottom example of figure 5.4 which are both located at local minima. Thus it seems hard to discover a generalization.

The selection of the points on the phase profiles had to be done manually. As far as we can tell, there is no obvious general way in which the selection could be done automatically, especially since the phase profiles in most cases will oscillate locally. Such oscillations occur despite the fact that the profiles have been smoothed in the $x$ direction. Possible improvements may be obtained by smoothing even further, although this should be done with caution so we do not lose necessary information about the points on the profile where disruption is actually caused by a true change in phase.

The strategy of estimating the phase value along the contour is very simple. If we were to suggest improvements, it would probably be wise to make an attempt to improve this estimate. The phase value at each point of the contour were estimated using the vertex and three additional points in the negative normal direction from the vertex. Instead of averaging between the phase values of the closest pixels to these points, the contribution of each point could be weighted. We could also choose to interpolate the phase values of the pixels close to a point, instead of merely using one pixel value per point.

Another point might be that we could exploit information that tells us in which proximity of the contour we expect to locate the clip points. Such information include placement of the probes, location of the interface and clip points in nearby image slices, and maybe even a baseline image defining the surface of the liver. This may limit the points on the contour down to a set of candidate points, among which we expect to find the right clip points. We have not followed up on these attempts to develop an automatic detection of clip points from the phase profiles as it seems unlikely that such a strategy would work, at least based on the available experimental data.
Segmentation based on a phase strategy
Chapter 6

Segmentation based on a curvature strategy in 2D

In this chapter we will investigate an alternative approach to segmenting the border between frozen tissue and air based on curvature information at the level set interface.

We begin our discussion on segmentation based on curvature measurements by studying a previous attempt to solve the problem. Heuch et al. investigated a strategy involving curvature in [8]. In this algorithm the first segmentation step was performed using snakes in 2D images. Although we performed the first part of the segmentation using the level set method on the whole image volume, i.e. in 3D, we may nevertheless test this strategy for the second part of the segmentation. This is done by extracting the contours from a slice of the level set function, i.e. we will use the approach from [8] on contours in two dimensions in a slice by slice fashion.

The curvature strategy does not make sense unless the curvature of the interface is changing as the interface reaches the surface of the liver, where we wish to clip it. We hence limit our study in this chapter, as well as the next, to cases where no other objects appear close to the liver and prevent the interface from bending at the liver surface, cf. section 5.2.

6.1 Curvature profiles

In the first part of this chapter we will investigate how curvature information along the contour from the first segmentation step can be exploited to perform the second part of the segmentation, i.e. the clipping of the level set contour. The contour consists of a set of vertices connected by straight
line segments. The second part of the segmentation problem is to separate the part of the contour that defines the border of frozen tissue from the part that has grown out of the liver and into the air above. We ask the reader to review section 5.2 for a more detailed description on the problem of clipping the contour, if needed.

We open this chapter by comparing the phase profiles from chapter 5 with profiles of a curvature scalar that will be defined below. Analogously to the previous chapter the notation here is consistent with the articles of Lobregt and Viergever ([15]) and Heuch et al. ([8]), where the curve is a discrete dynamic contour consisting of vertices in the plane connected by straight line segments (edges). The computations of geometric properties of their discrete contour model can be transferred to our level set contour which is extracted from the level set function. Such properties include local tangential and normal directions, curvature vector, and curvature scalar. In chapter 5 we established a local coordinate system at each vertex of the level set contour by defining a tangential and a normal direction at the vertex, see section 5.3. We now want to find a way to define the local curvature in a consistent manner. Since our contour model is discrete (it consists of points in the plane connected by straight line segments) the
6.1 Curvature profiles

Figure 6.2: In the figure above the curvature vector \( c_{i-1} \) points in the same direction as the normal vector \( \mathbf{r}_{i-1} \) and the curvature here is positive, while we have negative curvature at \( V_i \) where \( c_i \) points in the opposite direction of \( \mathbf{r}_i \). (The figure is redrawn from [15].)

Curvature is strictly speaking not defined at the vertices, while it is zero at the edges in between the vertices. We will nevertheless use the definition of local curvature at vertex \( V_i \) which is presented in [15]. The curvature vector is taken to be the difference between the directions of the two edges meeting at a vertex. More specifically we denote the local curvature vector at \( V_i \) by \( c_i \) and let

\[
c_i = \hat{d}_i - \hat{d}_{i-1},
\]

where \( \hat{d}_i \) and \( \hat{d}_{i-1} \) are the directional unit vectors of the edges that meet at \( V_i \). Figure 6.1 illustrates this. This definition of curvature is not to be confused with the real curvature of the implicitly defined level set contour. We stress the fact that we in this chapter stick to the curvature measure used in the mentioned articles, although this strictly speaking is not the curvature of the contour. Considering the curvature vector \( c_i \) in the local \( \hat{r}_i \hat{t}_i \)-coordinate system we observe that \( c_i \) either points in the same direction as \( \hat{r}_i \) or in the opposite direction, i.e. in the direction of \(-\hat{r}_i\). The length of \( c_i \) can be described by the dot product

\[
c_i = c_i \cdot \hat{r}_i,
\]
Figure 6.3: On the top left the curvature vectors are shown at each vertex of the contour in the image. The green labeled frames are shown in close-up: frame 1 on the top right, frame 2 on the bottom left, and frame 3 on the bottom right.

which will serve as a scalar curvature measure at vertex $V_i$. When the curvature vector points in the positive normal direction, the dot product in equation 6.2 will be positive, i.e. the curvature scalar $c_i$ is positive, whereas a curvature vector pointing in the direction of the negative normal vector yields a negative curvature scalar.

In this setting the contour plots will be functions $C = C(i)$, where the function value at $i$ gives the curvature scalar (as defined in equation 6.2) at the $i$’th vertex $V_i$. Again we wish to smooth the function in the $x$ direction, see equation 5.2 in chapter 5.

Figure 6.3 shows curvature vectors at each vertex along the level set contour. Notice that since the contour is locally jagged (not smooth), the curvature vectors point in both directions, for instance at the inner part of the frozen region. However, taking a small segment of the contour, the average curvature should represent the basic shape of the interface segment.
At the corners of the frozen region there is a majority of curvature vectors pointing outwards (from the contour), indicating a negative curvature at those locations. Likewise, the main tendency at the “brim” of the “mushroom hat”\(^1\) shows curvature vectors pointing inwards, reflecting positive curvature.

### 6.2 Results and discussion

We will initially show some curvature plots using the exact same example images as we did with the phase strategy in chapter 5, i.e. the level set interfaces are extracted from the PRF images.\(^2\) The purpose is to compare strategies on an equal basis. Secondly, we will use images from the 3D scan series. The latter images have finer resolution and no spacing between slices, hence they presumably serve better than the more coarsely gridded PRF images in accurately computing the level set interface and its curvature.

In the previous section we defined a local curvature scalar at each vertex of the level set contour in 2D. Plotting scalar profiles as a function of the vertex numbering we observe that the curvature oscillates fairly drastically. This observation agrees with what we would expect from studying the close-up image of the curvature vectors along the contour in figure 6.3: the contour is piecewise linear and therefore \textit{locally} not very smooth. This phenomenon may be caused by noise in the images. Noise can be a result of various circumstances, such as small errors (e.g. due to calibration) in the device that picks up and interprets the signal in the MRI scanner, or movement of the tissue due to freezing, etc. Another explanation behind the bad smoothness of the data may be that the images do not have high enough resolution to represent the variation in the physical state of the tissue. If the changes in the tissue happen at a more detailed level than the voxel size, the image data may naturally be noisy. Both these conditions may cause an irregular contour. Fortunately though, we see that there are some points in the profile, indicating that the curvature scalar suddenly makes a more drastic change of value here.

We made a script in order to locate the \textit{local minima} of the profile, and then select the \(n\) lowest values among the local minima values, where \(n\) is the number of clip points that we expect to find in the image. This simple strategy extracts the positions of the clip points in a completely automatic fashion.

\(^1\)The shape of the interface resemble a mushroom, cf. section 5.2.

\(^2\)The PRF images are acquired using a 2D-FGRE sequence, while the 3D images emanate from the 3D-SPGR sequence, cf. section 2.2.3.
Figure 6.4: Curvature profiles from PRF images. The image level set contour is shown in blue in the image, and the curvature values along the contour are plotted to the right of the image. Note that the red points in the image each corresponds to a point that is marked in the plot. Vertices are numbered clockwise starting from vertex $V_0$, which is indicated with a filled, white circle in the image.
Figure 6.5: See figure 6.4 for an explanation.
manner, except for the fact that we need to know how many points we must find, i.e. the number $n$ must be known. One should note that this strategy is not necessarily very robust. Since the curvature oscillates considerably along the contour, there are many local minimas. The challenge is to distinguish between the minimas that arise from noise and those that truthfully belong to the corner points of the frozen region. Thus the strategy is based on the assumption that the corner points lead to more distinct minimas than the rest of the points that are local curvature minima. The lack of robustness will be discussed more thoroughly below.

We will briefly mention a distinction between the way we locate corner points and the way this is performed in [8]. The algorithm in [8] performs an iterated search for candidate clip points along the contour. Each search begins at the vertex with the shortest distance to the heat sink of the cryo probe. Since there are two corner points for each contour, both a clockwise and an anti-clockwise search must be performed, i.e. one corner point must be found in each direction. A point is regarded as a candidate point for a true corner if the sum of up to eight consecutive vertices is less than -0.6, with the restriction that none of the vertices contributing to this sum have positive curvature. The curvature at a vertex is considered negative if the curvature is less than -0.04. (Remember that the curvature scalar is the length of the projection of the curvature vector onto the normal direction. From the plots we see that the curvature values in our examples lie between -0.8 and 0.6). In order to precisely identify the vertex representing the corner from the chain of up to eight vertices, the crossing point of the tangent vector of the first and last vertex of the chain is calculated. The intermediate vertex with the shortest distance to the crossing point is then selected as the corner vertex.

We performed tests using both algorithms in locating the corner points. Since we did not have access to data about the location of the heat sink in our experiments, we had to provide a manual selection of vertex point in order to start the search of the algorithm presented in [8]. The results showed no significant difference in performance between the two approaches. Thus we found our algorithm to be the most adequate, at least when the heat sink location is not known.

Curvature profile results are presented in the following two sections: the PRF images in section 6.2.1 and the 3D images in section 6.2.2. The routine for locating corner points at the border between ice and air is of course identical in both sections. The reader should note that although some of the discussion may belong in both sections, it will not be repeated.
6.2 Results and discussion

Figure 6.6: The contour of the frozen region, a cryo probe, and a corner point. The angle \( \alpha \) is the angle between the probe and the line segment defined by the corner point and the heat sink, and \( \beta \) is the angle between the latter line and the negative normal vector at the corner point. (The length of the vector is not unit in the illustration.) The dashed line illustrates a virtual plane perpendicular to the probe and through the heat sink. We expect all clip points to lie on the same side of the plane as the probe, i.e. \( \alpha < 90^\circ \). We also assume that \( \beta < 180^\circ \).

6.2.1 PRF images

Figures 6.4 and 6.5 show the results of applying this method to the same PRF images as we studied in chapter 5 when examining segmentation based on a phase strategy. The location of clip points based on the scalar curvature profiles were both easier to perform as well as more robust than using phase profiles. As the example plots show the clip points were all found correctly without manual intervention from the user. These are not enough examples to trust that the procedure will work in the general case though. In the next section we will single out some concerns about the curvature profile approach.

6.2.2 3D images

Results from applying the curvature profile approach on 3D images are shown in figures 6.7, 6.8 and 6.9. Although the approach works well in many of the test cases, we do encounter difficulties in others.

In some of the images (see the two upper images of figure 6.7) we notice that there is a small cavity inside the liver that borders to, but is not part of, the frozen area. The level set contour will experience additional points of strongly negative curvature at the edge of such a cavity. When one or both
Figure 6.7: Curvature profiles from 3D images. The level set contour is shown in blue in the image, and the curvature values along the contour are plotted to the right of the image. The upper image contains two separate contours, hence the two plots. Note that the red points in the image each corresponds to a point that is marked in the plot. Vertices are numbered clockwise starting from vertex $V_0$, which is indicated with a filled, white circle in the image.
Figure 6.8: See figure 6.7 for an explanation.
of the cavity’s corners have more negative curvature than a corner point of the frozen region, the automatic detection will fail. This may also happen even when there is no such cavity, but if there are other points along the contour with strongly negative curvature because of irregularities of the contour. The uppermost image of figure 6.8 illustrates that an irregularity of the contour may also lead to “false” detection of corner points. In this image the frozen region has four corners where clipping of the contour is necessary. However, only two of the four points that are found by the automatic method actually correspond to the desired solution.

These examples may possibly be solved by forcing the clip points to fulfill additional requirements, i.e. apart from being a local minimum in curvature. Again we adopt an idea presented in Heuch et al.’s article ([8]) on segmentation of frozen tissue. If we think of a virtual plane perpendicular to the cryo probe through the heat sink, it is assumed that the corner points of the frozen region are located on the same side of this plane as the probe, see figure 6.6. More specifically, the angle \( \alpha \), which is the angle between the outer end of the probe, the heat sink and the corner point, is required to be less than 90\(^\circ\). This requirement should be used with caution. If the frozen region also grows into air on the other side of the plane (perpendicular to the probe through the heat sink), the restriction obviously does not apply, observe examples in figure 6.8 (middle image) and figure 6.9 (top image). Yet another condition is that the angle \( \beta \), which is the angle between the line from the heat sink to the corner point and the negative normal vector of the boundary vertex, is less than 180\(^\circ\). Whenever these two conditions are not met, a candidate point should be discarded, and one must search for other possible locations to clip. We do not have access to the parameters needed for computing these angles in our images. However we have reason to believe that introducing such constraints for the position of the points would reject some of the troublesome points that we have discussed so far. Thus after rejection the method should continue its search for candidate clip points at the next ascendingly ordered local minima of the curvature profile.

In [8] a final step to control the candidate clip points is to compare the candidates from neighbouring image slices to each other. It is reasonable to demand that clip points in one slice agree in space with clip points in the adjacent slices. When this is not the case, the outlier should be rejected, and a new search must be made in order to replace the candidate point.

In the middle image of figure 6.8 we have a difficulty similar to the one pointed out above. The liver is not a homogeneous object when it comes to intensity, and may contain “holes” that do not belong to the frozen area. We see that the level set contour captures the border of a cavity residing on
Figure 6.9: See figure 6.7 for an explanation.
the surface of the liver. This may happen even if the cavity is not directly connected to the frozen area, since the contour grows out of the liver. Note that it does not make sense to request the angle $\alpha$ to be less than $90^\circ$ in this special case, since there are true corner points on both “sides” of the liver. The lower image in figure 6.8 shows an additional example of the contour having some points where the curvature is more negative than at the points we are looking for. Those two points are relatively close in space to the true corner points, thus the angle $\alpha$ (cf. figure 6.6) is probably less than $90^\circ$ for each of the points. The unwanted result is that erroneous candidate points for clipping will probably not become rejected as they should. We therefore find it questionable whether the given requirements on location of clip points are sufficient. Even if the approach works in the majority of the cases, robustness of the routine may be hard to obtain.

Figure 6.9 shows some of the cases in which the curvature approach succeeded. As registered on the plots, no extra constraints were needed on the clip points here. The segmentation method discussed in this section works especially well when the frozen region has a “nice” and regular shape. Examples of “nice” shapes are shown in the two lower images in the figure. These images are both taken from the same liver. When conducting this experiment (experiment 3, cf. chapter 2) the probes were placed relatively close together, and the frozen region accepted an “ideal” shape. The curvature profile approach did locate the right clip points in each and all of the image slices of this specific experiment.

As pointed out above, the curvature measure taken from [15] (see equations 6.1 and 6.2) is defined in what seems to be an ad hoc manner. Since the curvature of a contour is the limit of the change of the tangent vector to the curve as the spacing between vertices goes to zero, an intuitive way to define the curvature vector at vertex $V_i$ could be

$$c_i = \lim_{\|p_{i+1} - p_i\| \to 0} \frac{\hat{t}_{i+1} - \hat{t}_i}{\|p_{i+1} - p_i\|}.$$ 

Since we cannot calculate limits, the estimate must be used directly. Alternatively, we could have used the level set function $\phi$ defined at each voxel. By considering the voxels constituting one slice as a “flat” domain, i.e. making up a grid of pixels, the curvature of the implicit curve given by $\phi = 0$ in a 2D slice of the image volume is given as

$$\kappa = \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} = \frac{\phi_{xx} \phi_y^2 - 2 \phi_x \phi_y \phi_{xy} + \phi_{yy} \phi_x^2}{(\phi_x^2 + \phi_y^2)^{3/2}},$$

taken from [28]. In order to estimate the curvature at the vertices of the contour, we would have to interpolate over the grid of pixels, i.e. the
Figure 6.10: The histogram shows the success rate for the location of clip points in the different experiments.

voxels corresponding to the slice in consideration. Preliminary studies indicate that using equation 6.3 in the calculation of curvature profiles produces no significant difference from the results presented in this chapter. However, due to the limited time constraint of our project, we have not performed extensive research in order to test if (and if so, then also how) adjusting the definition of curvature would change the results of segmentation.

It is difficult to assess the success of the presented method. Røislien and Samset discuss how one can calculate a performance measure of a new proposed method, see [22]. They point out that testing new methods is complicated, especially when the true value to check the result against is not obtainable. Thus the best alternative is considered to be to check whether the new method agrees with existing methods. The example used in [22] is testing of a new automatic method that reads the size of a volume. Since the true reading is not known, the automatic method is checked against the manual work of a radiologist. Ideally, we would have tested our results against manual segmentation performed by an expert. A trained radiologist can manually select the boundary of the frozen region from the images. It would be interesting to test the amount of overlap this would give with the final boundary produced by our (semi-) automatic method. It is very time consuming to perform manual segmentation, and unfortunately we do not have data from a radiologist. Instead
we performed manual assessment of the clip points that were generated as outlined above. For each slice where the level set contour needed to be clipped, we compared the number of correctly located clip points \( c_i \) against the total number of clip points \( n_i \) in the slice, i.e. the success rate \( s_i \) in slice \( i \) was measured by

\[
s_i = \frac{c_i}{n_i},
\]

(6.4)

where \( c_i \) is an integer between 0 and \( n_i \). Since \( n_i \) is either 2 or 4, this yields \( s_i \in \{0, 0.25, 0.5, 0.75, 1\} \). Figure 6.10 shows a statistical report of the success rate of the method that locates clip points based on the slice images from the 3D scan series. Note that only the slices where the contour must be clipped are considered in this test, and the number of such slices varies between the experiments.
Chapter 7

Segmentation based on a curvature strategy in 3D

The idea that curvature measurements can be exploited in segmentation, especially of surfaces, is not new, see for instance [4, 5, 14, 17].

In the current chapter we will present our work regarding the extension of Heuch et al.’s curvature strategy in 2D (cf. chapter 6 and [8]), to our level set surface in 3D. Heuch et al. use a snake model that operates on 2D images. First of all, our contribution differs from this model by the fact that we perform level set segmentation on the 3D image volume. Secondly we utilize surface curvature instead of the curvature of a 2D contour. Moreover, our method is a generalization in that level set segmentation allows us to handle more general topological shapes of the frozen area. The primary objective in either case is however to recognize that the interface from the first segmentation step (whether it be a curve or a surface) will experience negative curvature as it extends from the frozen region and bends at the surface of the liver.  

7.1 Surface curvature

The aim of MR image acquisition during cryo ablation is to visualize the surgical impact on the organ structure inside the body. Ideally we would like to depict the ice-ball forming around the cryo probe(s) as a three dimensional object inside the liver, and not only observe it on a slice by slice manner. Therefore it is natural to ask oneself whether we can benefit from

---

1We hence limit our study to cases where no other objects located close to the liver prevents the interface from bending at the liver surface, cf. section 5.2.
studying the geometric properties of the level set surface of the whole image volume as an alternative to the collection of 2D contours from the set of separate slices. The level set algorithm operates in 3D and produces a level set function that defines a value for each voxel of the image volume. We may therefore obtain a triangulated surface representing the function's zero level set. In this way each point on the interface will utilize information collected at neighbouring points in various directions because of the extra dimension.

7.1.1 Mean, Gaussian and principal curvatures

The primary task is to compute surface curvature at the isosurface. The level set equation is solved numerically by dividing the spatial domain into grid points \((x, y, z)\) that hold approximations to the level set function \(\phi(x, y, z, t)\) at a given point \(t\) in time. Thus geometric properties of the interface such as curvature are easily calculated. When it comes to three dimensions, we have many choices for which curvature measure to use. In chapter 3 we derived the mean curvature \(\kappa_M\) of the level set surface as

\[
\kappa_M = \frac{\phi_x^2(\phi_{yy} + \phi_{zz}) + \phi_y^2(\phi_{xx} + \phi_{zz}) + \phi_z^2(\phi_{xx} + \phi_{yy})}{|\nabla\phi|^3} - \frac{2\phi_x \phi_y \phi_{xy} + 2\phi_x \phi_z \phi_{xz} + 2\phi_y \phi_z \phi_{yz}}{|\nabla\phi|^3}, \quad (7.1)
\]

as given in Sethian’s book [28]. The Gaussian curvature \(\kappa_G\) is another commonly used measure of how much a surface is bending, and it is given by

\[
\kappa_G = \frac{\phi_x^2(\phi_{yy} \phi_{zz} - \phi_{yz}^2) + \phi_y^2(\phi_{xx} \phi_{zz} - \phi_{xz}^2) + \phi_z^2(\phi_{xx} \phi_{yy} - \phi_{xy}^2)}{|\nabla\phi|^4} + \frac{2\phi_x \phi_y (\phi_{xz} \phi_{yz} - \phi_{xy} \phi_{zz}) + 2\phi_y \phi_z (\phi_{xy} \phi_{xz} - \phi_{yz} \phi_{xx})}{|\nabla\phi|^4} + \frac{2\phi_x \phi_z (\phi_{xy} \phi_{yz} - \phi_{xz} \phi_{yy})}{|\nabla\phi|^4}, \quad (7.2)
\]

again taken from [28].

The curvature of a surface may not be as intuitive to understand as that of a plane curve. The scalar curvature of a curve at a given point represents

\[\text{In previous chapters we have worked on the zero level isocontours obtained from the voxels consisting of a single slice, i.e. we have virtually operated on a level set function defined at each pixel of a 2D image.}\]
7.1 Surface curvature

Figure 7.1: Various types of curvature used as color maps.

the reciprocal of the radius of an osculating circle at that location. At a point where the curve bends a lot the osculating circle will have small radius \( r \), thus the curvature scalar \( 1/r \) will be relatively high, while the more flat a curve is, the greater the osculating circle and the smaller the curvature. The curvature \( \text{vector} \) at a given location of a plane curve is a vector pointing in the direction of the center of the osculating circle.

In order to understand the concept of surface curvature one may consider the intersection between the surface and a plane containing the surface’s normal vector at a given point. Imagine that we find the curvature of the intersecting curve at the point, and also for each of the other intersecting curves we get when varying the plane in consideration. We define the \textit{principal curvatures} \( \kappa_1 \) and \( \kappa_2 \) of the surface to be the maximum and the minimum values respectively of the scalar curvature of those intersecting curves. The corresponding extremal directions are called the principal directions, and they can be shown to always be perpendicular.

Gaussian curvature is the product of the principal curvatures,

\[
\kappa_G = \kappa_1 \kappa_2, \quad (7.3)
\]
while mean curvature is their average,
\[
\kappa_M = \frac{\kappa_1 + \kappa_2}{2}.
\]
(7.4)

Gaussian curvature is positive where the surface is locally convex or concave and negative where locally saddle. Mean curvature is an extrinsic property of the surface, i.e. it depends on the embedding of the surface in space. A plane has zero mean curvature, while if you roll it to form a cylinder, the mean curvature is nonzero. The Gaussian curvature is zero in both cases. The mean curvature of a sphere is constant and positive if the orientation of the surface normal is outwards, and negative if the normal points inwards. Gaussian curvature is an intrinsic property of the surface, meaning that it does not depend on the embedding of the surface in space. Intrinsic properties such as Gaussian curvature do not change sign when the direction of the normal vector of the surface is reversed, whereas the mean curvature does, [21].

Combining equations 7.3 and 7.4 we obtain formulas for the maximum (\(\kappa_1\)) and minimum (\(\kappa_2\)) principal curvatures
\[
\kappa_1 = \kappa_M + \sqrt{\kappa_M^2 - \kappa_G} \quad \text{and} \quad \kappa_2 = \kappa_M - \sqrt{\kappa_M^2 - \kappa_G}.
\]
(7.5)

The surface we are studying will often imitate the surface of a mushroom: The stalk of the mushroom is the frozen tissue inside the liver, having the shape of a drop around one cryo probe, or sometimes more drops grown together, when using several probes. The hat of the mushroom is the part of the interface that has grown out of the liver parenchyma, and thus the part that we wish to cut off in the second step of segmentation, i.e. after level set segmentation. Our ambition is to indicate a curve on the liver surface where the mushroom should be clipped. In other words, we must find the curve that lies on the surface and cuts the stalk from the hat. Points on this curve are likely to experience negative Gaussian curvature since they are saddle points, and also negative mean curvature assuming that the surface curves more in the minimum principal direction than in the maximum direction, i.e. that the absolute value of \(\kappa_2\) is greater than that of \(\kappa_1\). Note however that this observation is very simplified. The shape of the cutting boundary will often be more complex, not all boundary points need have negative curvature, and in the case of several probes, the boundary may even consist of two separate, closed curves.

In figure 7.1 we have estimated the different curvature measures for one of our 3D models. The triangulated mesh is extracted from the level set function volume, and the estimated curvature at each vertex is used as a color map on the surface.
7.1.2 Numerical computation of curvature

The curvature is calculated directly from the values of the level set function \( \phi \) using finite difference techniques as outlined in [19]. For the first partial derivatives we use a second-order accurate central difference, e.g.

\[
\phi_x = \frac{\partial \phi}{\partial x} \approx \frac{\phi_{i+1,j,k} - \phi_{i-1,j,k}}{2\Delta x},
\]

(7.6)

abbreviated \( D_0^x \phi \), and analogous formulas for \( \phi_y \approx D_0^y \phi \) and \( \phi_z \approx D_0^z \phi \). Second-order accurate finite difference formulas for the second partial derivatives are also suggested, e.g.

\[
\phi_{xx} = \frac{\partial^2 \phi}{\partial x^2} \approx \frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2},
\]

(7.7)

abbreviated \( D_0^x D_0^x \phi \), where \( D_0^x \) and \( D_0^y \) are the forward and backward difference approximations respectively to the first partial derivatives. \(^3\) The second partial derivatives \( \phi_{yy} \) and \( \phi_{zz} \) are defined similarly. Second-order accurate finite differences for the mixed partial derivatives are given by central differences, e.g. \( \phi_{xy} \approx D_0^x D_0^y \phi \).

How do we act when \( |\nabla \phi| = 0 \) which means that \( \kappa \to \infty \)? We limit the curvature based on a logical observation: Since the smallest unit of measure on our Cartesian grid is the resolution size of the pixels, i.e. \( \Delta X = \min\{\Delta x, \Delta y, \Delta z\} \), it makes little sense to hope to resolve objects smaller than this, e.g. spheres with a radius smaller than \( \Delta X \). The curvature of a circle with radius \( r \) is \( 1/r \) (see section 7.1.1), hence we limit the curvature \( \kappa \) so that \(-\frac{1}{\Delta X} \leq \kappa \leq \frac{1}{\Delta X}\), [19]. Whenever a calculated curvature value falls outside this range, we replace the value by the appropriate choice of \(-\frac{1}{\Delta X}\) or \(\frac{1}{\Delta X}\).

7.2 Surface curvature segmentation

The curvature at the vertices of our approximated isosurface should faithfully reflect the shape of the underlying surface. The precision of the curvature estimation will have great significance for the quality of the segmentation procedure.

Mesh segmentation algorithms exist in various forms and for a range of application purposes, but are typically performed by growing or merging regions incrementally, see [14] and references therein. The regions are

\(^3\)E.g. \( \phi_x \approx D_0^x \phi = (\phi_{i+1,j,k} - \phi_{i,j,k})/\Delta x \), \( \phi_y \approx D_0^y \phi = (\phi_{i,j+1,k} - \phi_{i,j,k})/\Delta y \), \( \phi_z \approx D_0^z \phi = (\phi_{i,j,k+1} - \phi_{i,j,k})/\Delta z \), and equivalently for the partial derivatives of the other variables.
Segmentation based on a curvature strategy in 3D

defined by clustering vertices, edges or polygons sharing similar features and then refining the border between these parts to find the segmentation. As an alternative to locating boundaries induced from finding regions, Lee et al. [14] propose a method where the boundary between sub-parts of the mesh is found directly.

According to cognitive theory the human perception naturally decomposes an object into its constituent parts by dividing it along lines of negative curvature minima. Various tests have been carried out that verify this behavioral tendency - known as the minima rule. Both [14] and [20] exploit this idea in their segmentation algorithms.

7.2.1 Watershed segmentation

Mangan and Whitaker present a method for segmenting a 3D surface mesh into meaningful, connected pieces in [17]. They base the partitioning of the mesh on surface curvature in that the patches resulting from the segmentation should have relatively consistent curvature throughout, and be bounded by areas of higher or drastically different curvature.

We propose a slightly altered version of their algorithm in which we use the negated curvature of the surface mesh, since we would like the boundaries between the segmented areas to represent a curve of strongly negative curvature, i.e. the curve cutting the mushroom “hat” from its “stalk”.

The idea behind the watershed approach stems from an image segmentation technique within mathematical morphology. We will first briefly present the original watershed transform, and next describe how it can be employed to partition our level set surface.

Morphological watersheds

In the field of digital image processing the watershed algorithm is a commonly used segmentation technique. The name of the algorithm arises from the way regions are segmented into catchment basins. Thinking of the graytone values defined over the two dimensional domain of the image as height values, any grayscale image can be thought of as a topographic surface with valleys and hills. If we start flooding each valley from its lowest point and prevent merging waters coming from different valleys, the

---

4By meaningful they intend that the relative size and organization of the segmented regions must be relevant to the application at hand. The solution will therefore depend heavily on one’s definition of meaningful.
7.2 Surface curvature segmentation

Figure 7.2: Black lines illustrates a height function. Initially there were three regions. Letting the water flood over the watershed lines we find which regions should become merged. If we decide to merge regions whose watershed depth is below an appropriate threshold, regions 1 and 2 would be joined together as illustrated.

surface will be divided into catchment basins separated by the ridges of the hills - also called the watershed lines.

The topography need not be specified by the image graytone itself, but can be given by a quite arbitrary height function. In fact the watershed algorithm does not depend on the height function. It can be any real valued function \( f \) defined on the image domain \( U \in \mathbb{R}^2 \), i.e. any function \( f(x,y) : U \rightarrow \mathbb{R} \). Of course the choice of \( f \) will affect the outcome of the segmentation and \( f \) should therefore be designed to fit the purpose of the application. Often the gradient magnitude of the image is used in order to locate regions that are relatively homogeneous. Figure 7.2 illustrates a height function and how flooding produces regions.

Watershed segmentation can be implemented by simulating the process of flooding regions, i.e. to find catchment basins by starting at each local minimum (bottom of a valley) and incrementally flood the region until it connects to a neighbouring region. This is called a bottom-up approach. An alternative is to take each point in the domain \( U \) and follow the path of steepest descent downward until reaching a minimum, or a point already associated with a minimum. Observe that a catchment basin is a set of points which all end up in the same local minimum of \( f \) when flowing downward like this. We have followed the latter approach in our
Figure 7.3: From above; a grayscale image, the magnitude of the image gradient, and the result of applying the watershed transform to the gradient image. The regions (separated by black lines) are far too many. Hence the flooding of the catchment basins should be followed by a merging of “shallow” regions.
7.2 Surface curvature segmentation

implementation, i.e. a top-down approach, cf. [17].

Performing only the steps described so far typically yield an oversegmentation of the image. Normally it is required to merge some neighbouring regions whose watershed depth (i.e. the height between the top of the ridge and the bottom of the valley) is small in order to obtain a useful segmentation.

Figure 7.3 illustrates the watershed transform applied to the gradient magnitude of a grayscale image. The gradient is often used as a height function when the goal is to partition the image into areas of similar grayscale depth. Typically we wish to locate the different objects in the image. In the example of figure 7.3 the result is clearly oversegmented. No merging was performed, thus the segmented image contains exceedingly many regions, and shows that a merging step is necessary.

Extension of the watershed method to 3D surface meshes

We assume that we wish to perform segmentation on a mesh $M$ that consists of a set of vertices $V$ connected by edges. We must define a scalar function $f$ on the mesh $M$, i.e. $f$ should assign a height value to each vertex in $V$. In general we could apply the watershed segmentation algorithm to any type of polygon mesh, but we will focus on triangulated meshes, thus the vertices constitute a set of triangle faces. We will divide the mesh $M$ into $R$ regions. A region is the equivalence of a catchment basin in the description of the watershed algorithm in the above section. Each region $M_r$ for $r = 0, \ldots, R - 1$ is defined by a connected subset of vertices from $V$. The regions must cover the entire mesh, and they must not overlap, i.e.

$$M = \bigcup_{r=0}^{R-1} M_r,$$

such that $M_r \cap M_s = \emptyset$ whenever $r \neq s$.

When applied to image processing the watershed algorithm operates on a rectangular grid in two dimensions where each grid point holds a value of the height function $f$. Extending to a surface mesh in 3D the counterpart to the grid points becomes the vertices of the mesh. Thus instead of following the path of steepest descent from a grid point along the grid lines in 2D, we must follow the edges of the mesh. Each vertex $v_i \in V$ is connected to a set of neighbouring vertices defined by the edges leaving $v_i$, or equivalently, the set of vertices that share a triangle (in the general mesh; share a face) with $v_i$. Thus the spatial position of the vertices will not influence the process of forming catchment basins. Only the topology of
the mesh and the definition of a height function at the vertices are relevant to the result of this process.

Based on the previous discussion, a natural choice of height function is a measure of curvature at the vertices. When the mesh arises from a known underlying implicit function, as is the case in our situation, the calculation of curvature is well defined. We have discrete function samples (one sample per voxel) from the implicit level set function \( \phi \), and compute the curvature numerically using finite differences as outlined in section 7.1.2. The curvature at the vertices, which are defined on a sub-voxel level, are then interpolated from the computed curvature values. Otherwise, in case the mesh does not arise from a function, or the function it approximates is unknown, an estimated discrete curvature measure may be employed, see e.g. [17, 31].

Studying the various 3D surfaces that arise from our experiments, and coloring them using curvature, such as in figure 7.1, we remark that the minimal curvature often provides the greatest color contrast at the boundary where we wish to clip the surface. Mangan and Whitaker construct their height function at the vertices using what they in their article call total curvature and define as \( \kappa_D = (4\kappa_M^2 - 2\kappa_G^2)^{1/2} \). According to Page et al. ([20]) Mangan and Whitaker propose a very robust segmentation strategy, but the method yields an ad hoc segmentation without support from a theory such as the minima rule (see discussion earlier in this section). In [20] it is therefore suggested to

..seek an algorithm that strictly adheres to the minima rule and yet leverages the robustness of a watershed segmentation.

Since the watershed method itself does not depend upon the height function \( f \), it is simple to experiment with various choices of \( f \), which is merely an input parameter to the segmentation. This makes it easy to experiment with segmentation using different types of curvature. We performed various such tests, and they confirmed what might already be suggested from the discussion above; that the minimum principal curvature serves well as a height function. When looking for the deepest valleys (instead of the highest ridges) to separate the regions of the mesh, we must use the opposite height function, i.e. instead of using the minimum principal curvature directly, we use the negative value of the curvature.

We attack the algorithm implementation problem by following nearly in the footsteps of [17]. The task we are facing is to assign a label to each vertex. The label indicates which region that the vertex belongs to. Thus connected regions of vertices that share similar curvature properties should eventually receive the same label. We will first define “flat regions”
as connected groups of vertices whose height function value differs by less than some small value $\epsilon$, which is a parameter defined in the program. A minimum is either a single vertex or a flat region whose neighbouring vertices are all “higher” as measured by the height function. (We use $\epsilon$ throughout and regard two values as equal unless they deviate from each other by more than $\epsilon$. This consideration is made due to uncertainty and noise in curvature estimations).

Our approach can be outlined as follows:

1. Compute a height function defined at the mesh vertices. More specifically we will define the height function to be the negative of the surface’s minimum principal curvature at the vertex, $-\kappa_2$

2. Assign a unique label to each local minimum.

3. Locate all other flat regions. (When a flat region is not a minimum, it is called a “plateau”).

4. Iterate the plateaus from step 3. For each plateau, follow the path of steepest descent until reaching a region that has a label. Then assign this label to the plateau where the descent started, as well as to all the vertices that were encountered along the path.

5. Allow each of the remaining unlabeled vertices of the mesh to descend and become included into the labeled region that they encounter.

6. Merge regions whose watershed depth is below a preset threshold.

7. Merge regions whose size - measured by its number of vertices - is below a preset threshold.

The first five steps of the algorithm should be rather self-explanatory, and we refer to Mangan and Whitaker’s article [17] for those interested in further explanation. We will however make a few notes concerning the merging process, as this is a substantial part of the algorithm, which leaves room for deciding upon application specific choices. We selected two merging strategies (see step 6 and 7) that we found suitable for our specific segmentation task.
Merging based on the watershed depth of the catchment basins (regions) is a choice that is natural from the way the algorithm has emerged. The algorithm is very sensible to even the smallest fluctuations in the height function (here: curvature) and will label every local minimum as a region of its own.

The watershed depth of a region is defined as the difference between the point in the region with the lowest value of the height function, and the vertex along the region’s boundary with the lowest height value of all other boundary vertices. This proportion reflects the greatest depth of “water” the region can hold before it “spills over”. During the first merging step we model what happens when some regions begin to overflow their boundaries. A shallow region (i.e. one whose watershed depth is below the preset threshold) will become merged with the neighbouring region that water would spill into. The region with which to merge is hence identified by locating the lowest boundary vertex and then choosing the steepest path of descent from this vertex.

The second merging step is perhaps more closely related to the application at hand, since we expect to obtain two-three rather large surface regions (depending on the shape of the liver and cryo lesion). Small regions may arise from “bumps” (at the surface) that are not negligible with respect to curvature, but that we do not want to constitute as an own region. Thus a final process that merges small regions into greater ones is desirable.

Usage

The watershed algorithm is very sensitive to irregularly varying surfaces. It might therefore be a good idea to smooth the data before applying the watershed segmentation algorithm. We exploited Slicer’s (cf. section 3.3) model maker module to extract and smooth the triangulated level set surface, as described in section 3.6, before we performed watershed segmentation. Parameters to the segmentation algorithm were chosen as follows:

- **Size merge threshold**: Our data varied in size with the number of vertices ranging from 12000 to 30000 (number of triangles from 24000 to 60000). The region size threshold was set to 2000 vertices.

Note that the boundary of a region may contain both vertices belonging to the region as well as vertices from its immediate neighbours. This is because a ridge on the mesh may contain vertices which have downhill neighbours flowing into separate catchment basins.
7.3 Results and discussion

- **Depth merge threshold**: In section 7.1.2 we explained why we restrict the value of the numerically computed curvature $\kappa$ to lie within the interval $[-\Delta X, \Delta X]$ where $\Delta X = \min\{\Delta x, \Delta y, \Delta z\}$. In our 3D images $\Delta X = \min\{\Delta x, \Delta y, \Delta z\} = 0.976562$, i.e. $\kappa \in [-1.024, 1.024]$. Thus the depth of a region can technically be between 0 and 2.05, but (especially the initial) regions are usually shallow with respect to this scale. Note that the minimum principal curvature values tend to lie in the lower part of the curvature scale. The region depth threshold was set to 0.19, so that regions whose depth is below 0.19 will become merged.

- **“Equal” values parameter**: Two vertices whose height function absolute difference is less than $\epsilon$ are treated as equal. We used $\epsilon = 0.005$. These parameters should of course be adjusted if the data on hand are different from ours.

7.3 Results and discussion

The results of watershed segmentation of the level set surfaces from our experiments are shown in figure 7.4. Each surface defines the border of the frozen region and a part of the space (air) outside the liver surface. As mentioned earlier, the level set interface does not stop at the border of the frozen region against air, since both air and frozen tissue are signal void in MR images. The result is that the surface contains a “hat” that must be separated from the part of the surface that belongs to the border of the ice-ball. Watershed segmentation yields a partition of the surface into regions, separated by ridges in the height function, i.e. contours of strongly negative minimum principal curvature. The goal was to obtain one region that represents the part of the surface that borders to the frozen area, and one (or two, in the case of the lower model in figure 7.4) region(s) that represent(s) the part of the surface that should become removed. As can be seen in figure 7.4, the proposed segmentation method succeeds in the creation of such regions.

The examples are taken from three different experiments. The examples that we have used represent a diverse set of shapes that the frozen region may accept. Comparing the shapes we see that each case is quite original. Thus so far our surface segmentation shows promising results.

One of the strengths of using watershed segmentation is that we are able to deside upon region merging criteria. Merging based on the depth of the catchment basins is probably a good choice in nearly all applications.
Figure 7.4: Result of watershed segmentation. The region labeled ice defines the border of the frozen region, while the regions labeled air represent the parts of the level set surface that have grown out of the frozen tissue and into the empty space outside the liver parenchyma. The edges of the triangles in the mesh are drawn in black.
Figure 7.5: The figure shows a model with colors indicating the minimum principal curvature value. Note that the surface has some small protrusions that are surrounded by curves of strongly negative curvature.

Figure 7.6: Watershed segmentation of the model from figure 7.5, each region is represented by a separate color. The left/right model shows the situation before/after performing merge based on region size. The edges of the triangles in the mesh are drawn in black.
where watershed segmentation is used. This is because even the smallest fluctuations caused by noise in the height function will lead to oversegmentation. Merging based on region size, however, is a choice that we made based on the fact that we did not expect to have any small regions in the surface in our application. In the previous chapter we experienced difficulties when the liver contained small cavities that did not belong to the frozen region. This led to confusion since the cavities are surrounded by curves of strongly negative curvature, just like the edges of the frozen region. Such cavities can be seen as protrusions on the level set surface, see figure 7.5 for an example. One of our challenges is not to mistake the edges of the small protrusions in the surface for being part of the edge of the frozen region. The segmentation based on curvature in 3D handles this problem well, since relatively small parts of the surface become merged into the surrounding region in the last step of the watershed segmentation algorithm, see figure 7.6.

7.4 Future work

One problem that remains to be solved is how to fill in the hole that is created in the level set surface if we remove the part that lies outside the liver surface. In the 2D case, Heuch et al. suggest to use a straight line segment connecting the corner points of the frozen region, after that the part of the 2D level set contour that is outside the liver has been removed, see [8]. Even if this may be an acceptable approximation in two dimensions, it does not allow for a straightforward extension to 3D. In the rest of this chapter we will discuss some thoughts around this problem, and show some preliminary work which may give a hint of how it should be solved.

7.4.1 Definition of the surface between ice and air

Let us first establish some notation in order to ease the discussion below. We will denote the complete boundary surface of the frozen area by $S$. (For now, we will not bother about how surfaces are represented, but rather think of a surface as a two-dimensional set in $\mathbb{R}^3$). Let $S_{\text{ice}} \subset S$ be the part of the surface $S$ that borders against normal tissue, and $S_{\text{air}} \subset S$ be the part of $S$ that borders against air. We have performed level set segmentation followed by watershed segmentation and thereby obtained a partition of the level set surface into different surface patches. One of those surface patches corresponds to a part of the boundary of the frozen region, thus
7.4 Future work

Figure 7.7: One surface region ($S_{cut}$) will be removed from the level set interface, and we are left with the surface region that borders to the frozen area ($S_{ice}$). The black contour indicates the incision which defines the boundary of $S_{ice}$.

it is the surface which we have called $S_{ice}$, while the remaining patch (or patches) correspond(s) to the part of the level set surface that has grown out of the liver, and will be denoted $S_{cut}$. Note that $S_{cut} \not\subset S$.

So far, we have used a triangulated mesh in order to represent the level set surface. Removing a part of this mesh, we are left with an open surface (i.e. homeomorphic to a bounded plane in 3D), when what we want is a closed surface (i.e. homeomorphic to a sphere). The boundary of the given surface defines a “hole”, although not in the topological sense of a hole, but a hole preventing the surface from being closed. All we know is that we wish to cover the hole with a fairly flat surface patch, $S_{air}$, which approximates the frozen part of the outer face of the liver. We know where the boundary of $S_{air}$ should lie, since it must be joined with the boundary of $S_{ice}$, see figure 7.8. We also remark that $S_{ice}$ and $S_{air}$ are not necessarily smoothly joined. In fact, we expect the boundary curve between the two surface patches to consist of points where the curvature of the joint surface $S$ changes strongly in the positive direction.

An initial idea is to find $S_{air}$ by triangulating the hole. However, it is not straightforward how this should be done. One difficulty is that the edges forming the boundary of the hole point in random directions, and naive triangulation of such holes often yields self-intersecting geometry, see [2]. We can not assume that the boundary of the hole is convex, we have edges that must be constrained in the triangulation, and we have uneven distribution of points, i.e. the point density at the boundary of the
Figure 7.8: The above figure illustrates the surface $S_{\text{ice}}$ which contains a hole, defined by a boundary contour (the solid black contour in the figure). We want to find the surface $S_{\text{air}}$ that can cover this hole. The point cloud consists of the points from the removed surface $S_{\text{air}}$ that are assumed to lie on the surface of the liver, and will therefore be used in the approximation of the surface inside the hole.

hole is large, while there are no points known to lie inside the hole.

Instead, let us assume that we would like to approximate $S_{\text{air}}$ by a parametrized quadratic polynomial surface $P : \mathbb{R}^2 \rightarrow \mathbb{R}^3$,

$$P(u, v) = \vec{c}_0 + \vec{c}_1 u + \vec{c}_2 v + \vec{c}_3 u^2 + \vec{c}_4 v^2 + \vec{c}_5 uv,$$  \hspace{1cm} (7.8)

where the coefficients $\vec{c}_i \in \mathbb{R}^3$.

The polynomial surface should be adapted to fit the hole of the surface $S_{\text{ice}}$. The hole can be described by the boundary of the corresponding surface region from the watershed segmentation, see figure 7.8. We let $V_1$ denote the ordered set of vertices that represent the boundary. For each vertex $v_i$ we know its coordinates $(x_i, y_i, z_i)$ in space. Since $S_{\text{air}}$ is the part of the surface of the liver that is frozen, it is a good idea to use points that we know belong to the liver surface in the approximation. Note that the surface patch $S_{\text{cut}}$ contains some points from the surface of the liver, since it represents the part of the level set interface that has grown out of the frozen region, and therefore folds out over the outer face of the liver (and reaches into air). We create a subset of points $(x_i, y_i, z_i)$ from this surface by only including points that lie on the surface of the liver. Let this point set be denoted by $V_2$. In order to know which points from $S_{\text{cut}}$ that should be included in $V_2$, we check the magnitude of the MRI signal in a small neighbourhood of each point of $S_{\text{cut}}$. We assume that points where the
Figure 7.9: We use Principal Component Analysis in order to make a parametrization of the points \((u_i, v_i) \rightarrow (x_i, y_i, z_i)\). The figure shows the projection of the points in the point cloud in figure 7.8 projected onto the parameter domain. The points corresponding to the boundary of the frozen region surface are connected with a red contour.

Figure 7.10: A first try to approximate the surface covering the hole by using a parametric quadratic polynomial surface.
magnitude is high represent liver tissue, and hence they should be in \( V_2 \), unlike points where the signal strength is low, and therefore are assumed to represent air, see figure 7.8. Although the points in \( V_2 \) do not belong to \( S_{air} \), they represent a surface that is smoothly connected to \( S_{air} \).

We wish to find the polynomial \( P \) given by equation 7.8 that best approximates the points \( V_1 \cup V_2 = \{(x_i, y_i, z_i)\}_{i=0}^{n-1} \) in the least squares sense. First, we need a parametrization of points \((u_i, v_i) \rightarrow (x_i, y_i, z_i)\) for \( i = 0, \ldots, n - 1 \), ideally one that preserves the topology of the boundary of the hole (no self-intersections). We may assume that the points \((x_i, y_i, z_i)\) are sampled from a relatively flat and smooth surface, hence one choice is to use principal component analysis in order to find the plane that best separates the points along the coordinate axes, and use the projection of the points onto this plane as parametrization. Figure 7.9 shows the projection of the point cloud from figure 7.8 using this type of parametrization.

The task is then to minimize the error of the approximation \( P(u_i, v_i) \approx (x_i, y_i, z_i) \) by finding the polynomial \( P = P(u, v) \) that minimizes the sum of least squared errors for the given points, i.e.

\[
P = \min_{Q \in \Pi_2} \sum_{i=0}^{n-1} \|Q(u_i, v_i) - (x_i, y_i, z_i)\|,
\]

where \( \Pi_2 \) is the space of all parametric quadratic polynomials defined on the parameter domain, which is \( \mathbb{R}^2 \), and \( \| \cdot \| \) is the Euclidean norm. Figure 7.10 shows the parametric surface that solves the minimization problem in equation 7.9 for the same example as used previously in this section. The triangulated mesh of the frozen region, whose hole we are trying to cover with the approximated surface, is shown in the same figure.

Since the surface that we seek to approximate should be smooth and relatively flat, we might obtain better results if we add a smoothness constraint. This can be done for instance by turning it into a so-called penalized least squares minimization problem, hence we find the polynomial \( P \) that minimizes

\[
P = \min_{Q \in \Pi_2} \sum_{i=0}^{n-1} \|Q(u_i, v_i) - (x_i, y_i, z_i)\| + \lambda D^2(Q),
\]

where \( \lambda > 0 \) is a constant, and \( D^2(Q) \) is a measure of the second derivative of \( Q \). Thus what we have done is to add the term \( \lambda D^2(Q) \) to the expression 7.9. Minimizing the new expression 7.10, we seek to find the surface \( P \) where not only the sum of the squared distances from the surface to the points in the approximation should be small, but we also want the second
derivative of $P$ to be small, which basically means that we desire a surface that bends as little as possible.

In addition, we must find out how to confine the approximated surface, represented by $P$, so that it can be connected to the boundary of $S_{\text{ice}}$. If the approximation to $S_{\text{ice}}$ is a parametrized polynomial surface, as outlined above, the easiest is probably to extend $S_{\text{ice}}$ smoothly so that the new boundary of $S_{\text{ice}}$ becomes the curve of intersection with the approximated surface. Then $S_{\text{air}}$ is the part of the surface represented by $P$ that lies inside this curve of intersection.

Alternatively, we can approach the approximation in a different manner. Perhaps a better idea is to define a partial differential equation in which the unknown function is a representation of $S_{\text{air}}$. Using this approach, we do not make any assumptions saying that the approximation must be a specific type of function, e.g. that it must be a polynomial. What we have in mind is a differential equation that tries to minimize some kind of energy measure over the unknown surface. The energy over the surface is typically defined as a function of the second partial derivatives of the function that represents the surface. Defining a grid on the domain for the unknown function, the total energy over the surface is calculated as a sum over the grid points. In this way we obtain sufficiently many unknown parameters in the equation in order to add interpolation constraints, i.e. we let the points on the boundary of $S_{\text{ice}}$ be given as boundary conditions.

Which method to use, remains an open question that we leave for further research. See also chapter 8 (Summary and conclusion), where we further address the topic of discussion in the current section.
Segmentation based on a curvature strategy in 3D
Chapter 8

Summary and conclusion

In the current chapter we will briefly restate the main problems we have considered in this thesis, and sum up how we attempted to solve them. We will then recapitulate the most prominent results, positive and negative. Last, but not least, we will present what we find to be the most interesting of the ideas for further research.

In this thesis we have attempted to develop a method for segmenting the frozen region in MR images acquired during cryo ablation of hepatic cancer tumors. The border of this region must be provided as an initial condition to the differential equations that can be solved to find the temperature distribution inside the frozen area. We performed the segmentation in two main steps.

In the first step we separated the ice-ball from the unfrozen tissue, using level set segmentation, and pre-existing software (Slicer), which implements the level set method for segmentation of medical images. During this process we exploit the fact that normal tissue appears as bright areas of high intensity, while the frozen tissue gives dark regions in the images. The level set interface is defined by solving a partial differential equation which depends on space and time. The segmentation process begins by initializing an interface as a small sphere centered at the probe tip inside the frozen region, and is then evolved in time. Suitable stopping criteria are implemented in order to halt the evolution as the interface reaches boundaries of high magnitude in the image gradient, and another term controls the smoothness of the interface.

Our studies show that application of level set methods to solve this task provides proper and robust segmentation. Compared to previous work based on segmentation by snakes, we have established that the level set methods are more flexible with respect to the topology of the frozen region, and therefore able to handle a variety of shapes. It is moreover an
advantage to perform the segmentation process in 3D. Although the snake model may be extended for use in three dimensions, it would require a more complicated method than what is offered by the implicit representation of the level set framework.

The second step of the segmentation was aimed at separating frozen tissue from air. Both substances are signal void in the MR images, hence additional information is necessary in order to separate them. Since the level set algorithm cannot separate frozen tissue from air, we are left with an interface that only partially defines the boundary of the ice-ball. The second part of the segmentation process thus consists in finding the remaining part of the boundary of the frozen area, i.e. the part that borders against air. This means that we have to

1. locate where the level set interface must be clipped (locate clip points of the level set contour in 2D / locate clip contour(s) of the level set surface in 3D). We want to keep the part of the interface that belongs to the frozen region only, i.e. not the part that belongs to air.

2. define how to complete the interface that is kept from the above step. This involves defining a contour between clip points in 2D (in previous work it has been suggested to simply use straight line segments between each pair of clip points), or a surface patch to fill in the hole that is bounded by the clip contour in 3D.

We have focused on the first of those two steps. For this part of the process we investigated three different approaches, each of which will be discussed below.

**Segmentation based on a phase strategy**

We presented segmentation based on a phase strategy as a promising idea, that unfortunately did not lead to success. We know that the temperature at the border of the frozen region is approximately constant, and that the temperature gradually increases as we move away from the ice-ball. We also know that the level set interface follows this border inside the liver, while it bends away from the frozen region at the surface of the liver. Since the phase of the complex valued MRI signal is closely related to temperature, we wanted to study the phase values at the location of the level set interface, and expected to find a significant change of phase values at points of the interface that correspond to the edge of the frozen region. Although the phase did undergo a change here, the type of change was unpredictable. Would it be a local minimum or a maximum? Also, we
noted that the phase changed significantly also at other locations where the geometry of the level set interface changed, i.e. we did not find a special relation between phase change and temperature change.

In order to make use of the phase information along the level set contours, we had to unwrap the phase images. The MRI scanner wraps the phase of the signal to the interval $[-\pi, \pi)$, hence phase unwrapping is necessary to reverse this process and obtain what is believed to be the true phase values of the signal. We proposed a phase unwrapping method that works well under the assumption that the true phase difference between adjacent pixels is no larger than $\pi$ in absolute value. For heterogeneous objects this is a fair assumption to make. An interesting extension of our method could be to unwrap images containing several types of objects, especially the type of images we would encounter in clinical situations where cryo ablation is performed.

**Segmentation based on a curvature strategy in 2D**

We tested a previously presented method on our image data, utilizing a curvature measure for discrete contour models in 2D, suggested by Lobregt and Viergever in [15], and applied in order to locate clip points by Heuch et al. in [8]. With a curvature profile we understand a function of ordered vertex indices along the contour with function values equal to the scalar curvature values at the vertices. Ideally, the curvature profile would be a well behaved function that changes continuously and experience local extrema where the contour bends accordingly. In the real world there are many reasons why the profiles may not be smooth:

- The image data contain noise.
- The resolution of the image is too coarse to re-create the shape of the contour at detailed levels.
- The given curvature measure is not feasible.
- The image data are not smoothed enough prior to level set segmentation.
- The level set function has not been smoothed.
- The curvature values are not smoothed enough or a different smoothing filter should be applied.
It is hard to do anything about the first two factors, but the remaining items in the list are all worth a closer study in order to improve the strategy.

The results of segmentation based on the 2D curvature strategy showed varying success. Our studies indicate that the strategy is a good alternative in cases where the frozen region has the “ideal” (mushroom) shape, with distinct corners of the frozen region, and no significant deformations, e.g. holes, in the tissue. In the general case, however, we experienced that additional constraints - besides being a curvature minimum - must be forced upon the clip points. We have discussed some ideas in chapter 6, but implementation, testing and further development will need more work.

**Segmentation based on a curvature strategy in 3D**

We proposed a method that performs segmentation of the frozen region in three dimensions. This is both natural since the level set segmentation is performed in 3D, as well as more effective. The goal is to obtain a 3D model of the frozen area and the temperature within. Previously proposed methods perform the segmentation in 2D, and clip points are found in each image slice, hence the information from the different slices has to be tailored together in the end. Our method relieves the segmentation task by instead seeking to locate the clip contour directly on the level set surface.

Morphological watersheds, which is an image segmentation technique, was generalized to 3D surfaces by Mangan and Whitaker in [17]. This method partitions a surface mesh into regions based on a so-called height function defined at the vertices, e.g. based on curvature. With no modification, their method lacked robustness when applied to our data. With inspiration derived from other work on surface mesh segmentation ([20] and [14]), we adjusted the method of Mangan and Whitaker in order to accommodate our problem. The main difference from their method is that we use the minimum principal curvature in our definition of the height function. The resulting method partitions the surface into regions separated by curves of negative curvature minima. We obtained promising results with this type of segmentation, which seems to be fairly robust, provided that we know enough about the data to supply the adequate parameters.

The partitioning of the triangle mesh yields surface regions separated by boundaries of ordered vertices, connected by straight line segments as edges. We remove the surface regions that are not bordering against the frozen region. The remaining surface does not completely define the border of the ice-ball, since it lacks the border between ice and air at the surface of the liver. We need to find a surface that defines this border, and that can be joined with the surface we already have. The boundary of the
known part of the surface is the contour of negative curvature minima that separated the frozen surface region from the discarded regions during watershed segmentation. It is reasonable to assume that the contour should be smooth, and not a ragged, piecewise linear edge, as is what we get from the partitioning of the triangle mesh. We may suggest an improvement by using our contour as an initial guess to a deformable contour model (using snakes or level sets) defined on the mesh. In light of the strength of level set segmentation, we might even consider to use these methods instead of watershed segmentation. Instead of solving the level set equation on a rectangular mesh with the image intensity as an outer “drive” to the evolution, the equation could be solved on a triangulated mesh with the curvature function used in the definition of the speed function. This is a task for the future.

In this thesis we have provided some thoughts and ideas around solving the problem of how to define the missing part of the surface. However, we think the main challenge in segmentation of the frozen region was to find out where the level set interface had to be clipped. Normally the boundary of the frozen region is not needed for visualization purposes, only for giving the proper boundary condition to the temperature equations. Future work should focus on adjusting the representation of output from our algorithm to be consistent with the needed input to the algorithm that calculates temperature. Since the latter algorithm only needs a labelmap volume (i.e. each voxel must be given a label indicating whether it belongs to frozen tissue, healthy tissue, air, metal (cryo probe), or other material), perhaps there is no need for a very refined definition of the missing part of the surface. All we need is an approximation method that finds the correct interface on a voxel level of detail.
Bibliography


[8] Håkon Heuch, Per Kristian Hol, Sumit Roy, and Eigel Samset. Automatic segmentation of frozen tissue from MR-images during cryo ab-
lation of the liver. Unpublished article. The Interventional Centre, Rikshospitalet University Hospital, Oslo, 2005.


