A Gaussian Mixture Model based Level Set Method for Volume Segmentation in Medical Images

Department of Mathematics, Linköping University

Grayson Webb

LiTH-MAT-EX–2018/08–SE

Thesis: **30 hp**

Level: **A**

Supervisors: **Erik Edespong**,
Sectra

Martin Singull,
Department of Mathematics, Linköping University

Examiner: **Fredrik Berntsson**,
Department of Mathematics, Linköping University

Linköping: **June 2018**

Abstract

This thesis proposes a probabilistic level set method to be used in segmentation of tumors with heterogeneous intensities. It models the intensities of the tumor and surrounding tissue using Gaussian mixture models. Through a contour based initialization procedure samples are gathered to be used in expectation maximization of the mixture model parameters. The proposed method is compared against a threshold-based segmentation method using MRI images retrieved from The Cancer Imaging Archive. The cases are manually segmented and an automated testing procedure is used to find optimal parameters for the proposed method and then it is tested against the threshold-based method. Segmentation times, dice coefficients, and volume errors are compared. The evaluation reveals that the proposed method has a comparable mean segmentation time to the threshold-based method, and performs faster in cases where the volume error does not exceed 40%. The mean dice coefficient and volume error are also improved while achieving lower deviation.

Keywords:

Probablistic level set methods, Gaussian mixture models, image segmentation, volume segmentation, medical images.

URL for electronic version:

http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-148548

Acknowledgments

First and foremost I would like to thank my supervisors Erik Edespong and Martin Singull for providing invaluable guidance, encouragement, and feedback throughout the process of writing this thesis. I would also like to thank my examiner Fredrik Berntsson for providing quick feedback and helping to ensure the thesis could finish on time. Furthermore, I would like to thank my opponent Axel Tiger Norkvist for his thorough reading of the thesis, and the constructive criticism provided. Lastly I would like to thank my family and friends for all their support and encouragement. It means the world to me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

When solving differential equations it is common to divide the domain into a discrete set of points and then evolve these according to the differential equation. In one dimension one might consider a wave equation defined on an interval. A certain number of points on the interval are chosen and some form of time stepping is employed (e.g., Runge–Kutta methods) together with finite differences in the spatial domain. To visualize the wave at a later time point, line segments are drawn between the values of the wave at the chosen points. Assuming step sizes are chosen appropriately, stability is guaranteed and error sizes can be controlled.

Now consider an interface evolving under some velocity field. In two dimensions this corresponds to the boundary (curve) of a connected region and in three dimensions it corresponds to the boundary (surface) of a connected volume. As earlier, a first approach is to discretize the interface into a finite number of points. In two dimensions these can be connected through line segments and in three dimension by triangles (i.e., a mesh). While this approach may work for cases where the topology does not change much under the velocity field, for example a circle that is evolved under a velocity field pointing out with speed one along the unit normal, it is easy to construct velocity fields that cause large distortions. To avoid the accuracy deteriorating one would have to periodically modify the discretization to account for these distortions by regularizing deformed interface elements. While such methods have been constructed, see for example [14], there is a simpler way to handle such difficulties, namely the *level set method*.

The level set method was introduced in 1988 by Osher and Sethian, [6], for solving differential equations related to curvature dependent flow. To understand the idea behind the level set method, consider a two-dimensional curve. Let $\phi(x)$ be a function defined on $\mathbb{R}^2$ in such a way that the zero level set of $\phi(x)$ corresponds to the curve and the points where $\phi(x) < 0$ correspond to the interior while $\phi(x) > 0$ corresponds to the exterior. While there are of course more details to consider, this is the core method. As we will see this will allow us to transform the curve's parametrized differential equation into one that can be evolved using standard methods described above. Once evolution of ϕ has completed it is only necessary identify its zero level set to observe the evolution of the curve.

The generality of this technique has made it applicable in a diverse set of

fields. Many of these involve simulations and computational physics, such as compressible flow, shock waves, solid-fluid interactions, low-speed flames, and heat flow. Another field where use of level set methods has become prevalent is image processing, such as reconstruction of surfaces from unorganized data, motion analysis, and the focus of this thesis; image segmentation. See [18], [16], and [22] for surveys of the various applications.

The goal of image segmentation is to determine the boundary of some feature. Methods that attempt to achieve this are called *Active contour models* and their starting point is often to minimize some form of energy that depends on the segmentation. Geodesic active contours, introduced in [11], compute an energy along the boundary of the segmentation that is inversely proportional to the gradient of the image. In ideal cases this finds the desired contour of the image feature as the gradient of the image is high around the feature's border. However, as many images contain noise and internal edges of the feature, this method can fail as it is attracted to local minima. While the effect of noise in an image can be partially negated by smoothing the image, this in turn can cause edges to blur and the method to fail at identifying the feature's boundary. When this happens the segmentation might not stop when it reaches a blurred edge, causing the segmentation to include too much of the image in the segmentation. This is commonly referred to as *leaking*.

Medical images are a prime example of when geodesic active contour models can fail. Transitions between internal structures are usually smooth and the region of interest is identified by intensity differences compared to its surroundings rather than edges, see Figure 1.1. This has given rise to *region based* active contour models. These aim to identify statistical properties of the feature in order to distinguish it from the background. The first such method was developed by Chan and Vese in [15]. The model they derived considers means of the interior and exterior region and uses this information to propagate the interface. Much work has been done to improve and build upon region based image segmentation and a good review can be found in [22]. Many Chan–Vese inspired region based models, however, assume that the feature being segmented is homogeneous. In this thesis a level set method for segmenting heterogeneous objects with weak boundaries is proposed.

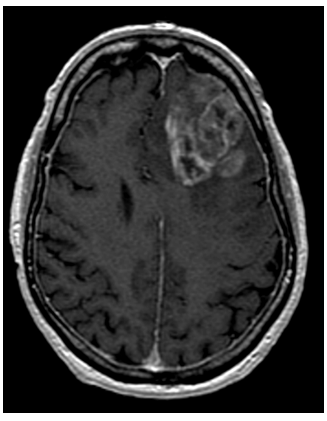


Figure 1.1: An MRI image where tumor boundary has regions of low gradient magnitude.

1.1 Motivation and Purpose

In this thesis we will restrict our attention to medical images taken by magnetic resonance imaging (MRI). The procedure to create such images involves measuring magnetic fields of the body and converting these measurements into gray scale images where intensity differences distinguish structures. The result is a volumetric dataset detailing the intensities measured in the region of body scanned. Furthermore we will be looking at MRI images taken of brains containing tumors. These are detected in the images by giving the patient contrast medium, such as Gadolinium, binded to glucose. Since tumors are a collection of cells with uncontrolled growth they will consume more of the glucose and thus also contrast medium than the surrounding tissue, resulting in an increased intensity in the MRI image where the tumor is located. It should be noted that there are many different types of MRI images that can be taken. We will be looking at so-called T1 weighted images in which fat tissue is highlighted. Furthermore, MRI images are 16-bit images while computer screens can only display images in 8-bit. In practice this means that when viewing images one must choose a *window level*. This specifies a center intensity and a window width. The intensities are then scaled to 8-bit according to their location within the width of intensities considered, and if they fall outside the window width they are clamped to black or white.

Parts of the tumor may not receive enough nutrients to stay alive as the tumor tissue closer to the region's border consumes an abnormal amount, resulting in cell death. Such regions are commonly referred to as *necrotic* and will appear as dark in the T1 weighted MRI image. This phenomena can occur to a varying degree and thus the tumors are often very chaotic both in topology and MRI-intensity, see Figure 1.2.

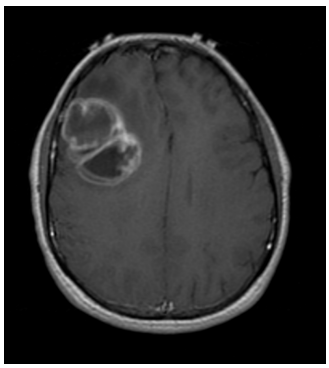


Figure 1.2: A T1 weighted image with a tumor containing necrotic tissue which shows up as dark on the MRI image.

Sectra AB is a company active within medical technology (and encrypted communication systems). Their picture archiving and communication system (PACS) is used by radiologists (among others) to analyze medical images and the software has a tool for real-time 3D volume segmentation based on a variant of the previously mentioned Chan–Vese active contour model. By real-time we mean that it is able to segment most structures in less than a second and has a simple user interaction (i.e., quick setup). The method segments homogeneous

tumors well but for more complicated cases such as the aforementioned heterogeneous tumors, it has difficulty producing an accurate segmentation. This thesis aims to propose and test an alternative method that is able to better handle heterogeneous tumors. The goal will be to answer the following questions:

- How is the level set method used for image segmentation?
- Can the level set method be used to develop a segmentation algorithm that performs better on MRI images of heterogeneous brain tumors than the existing method?
- How does segmentation time for the new method compare to the existing method?

The theory needed to understand the current method and the implementation of the new method, as well as the method for testing the performance, will be explained in the following chapter. In Chapter 3 the method for implementing, testing, and evaluating the proposed segmentation algorithm will be explained. In Chapter 4 and 5 the main results of this thesis are presented and interpreted. We end with some final conclusions in Chapter 5.

Chapter 2

Theoretical Background

This chapter begins by discussing the general level set method. This is followed by the derivation of an algorithm for computing the maximum likelihood estimators of parameters for a mixture of Gaussian distributions. Thereafter the theory for image segmentation using level set methods through the derivation of the Chan–Vese level set equation is covered. We then present the level set equation for the current method we wish to improve, and derive the proposed level set method. We end by seeing how efficient numerical methods can be designed to speed up computations.

2.1 The Level Set Method

2.1.1 Representation of an Interface

As the purpose is to develop a method for volume segmentation we will mainly focus on surfaces in three dimensions. Note however that the level set method is general and can be applied in any dimension. The two dimensional interface located in three dimensions is a closed surface that partitions $\mathbb{R}^3$ into separate subdomains with nonzero volume.

The surface can be represented explicitly or implicitly. When representing the surface explicitly, some form of parametrization must be constructed. For all but the most trivial cases this can be a cumbersome task, if not impossible. Discretization of a surface can be quite difficult as well. In two dimensions the connectivity of a curve is easy since it is determined by the ordering of the points chosen to discretize the curve. For a surface it is necessary to choose a number of points and specify their connectivity. Such a representation is further complicated when deforming the surface which can cause connectivity to change. This is why we use implicit level set representation.

To illustrate the process let us consider a simple example, the unit sphere $\Omega = \{\vec{x} \in \mathbb{R}^3; |\vec{x}| \leq 1\}$. Explicitly we may write its boundary $\partial\Omega$ as

$$S(\alpha, \beta) = \begin{pmatrix} \cos(\alpha)\sin(\beta) \\ \sin(\alpha)\sin(\beta) \\ \cos(\beta) \end{pmatrix}, \quad \alpha, \beta \in \mathbb{R}. \tag{2.1}$$

On the other hand we know that the unit sphere describes all points $\vec{x} \in \mathbb{R}^3$

that fulfill the equality $|\vec{x}|^2 = 1$. Letting $\vec{x} = (x, y, z)^T$ we can define

$$\phi(\vec{x}) = x^2 + y^2 + z^2 - 1. \tag{2.2}$$

Now $\partial\Omega$ described explicitly by (2.2) is represented implicitly as the zero isocontour of $\phi(\vec{x})$. The interior of Ω, denoted by Ω^- is given by $\vec{x}$ where $\phi(\vec{x}) < 0$, and the exterior Ω^+ are all points satisfying $\phi(\vec{x}) > 0$.

2.1.2 Geometric Tools

We will need some tools in translating expressions given in parametric form to their implicit equivalent. To begin let us consider a general level set function $\phi(\vec{x}) : \mathbb{R}^3 \to \mathbb{R}$, where the interior region of the volume surface, denoted Ω^- is given by $\phi(\vec{x}) \leq 0$, and the boundary by $\phi(\vec{x}) = 0$.

A well known fact from multi-variable calculus is that the gradient is perpendicular to the isocontours and points in the direction of increase. For our level set function it means that the outward facing unit normal of the interface $\partial\Omega^-$ is given by

$$\vec{N} = \frac{\nabla\phi}{|\nabla\phi|}. \tag{2.3}$$

Note that the normal is defined on all of $\mathbb{R}^3$ where $|\nabla\phi| \neq 0$,while coinciding with the explicit representation's normal for points located on the interface. The curvature κ of the interface is defined as the divergence of the normal, and so using (2.3) we get

$$\kappa = \nabla \cdot \vec{N} = \nabla \cdot \left(\frac{\nabla\phi}{|\nabla\phi|} \right). \tag{2.4}$$

We now define the characteristic function for the interior region including the boundary, $\overline{\Omega^-}$ as

$$\chi^-(\vec{x}) = \begin{cases} 1 & \text{if } \phi(\vec{x}) \leq 0, \\ 0 & \text{if } \phi(\vec{x}) > 0. \end{cases} \tag{2.5}$$

Likewise, the characteristic function for the exterior region Ω^+ is defined as

$$\chi^+(\vec{x}) = \begin{cases} 1 & \text{if } \phi(\vec{x}) > 0, \\ 0 & \text{if } \phi(\vec{x}) \leq 0. \end{cases} \tag{2.6}$$

The one-dimensional *Heaveside* function can alternatively be used to define the characteristic functions, and it is given by

$$H(\phi) = \begin{cases} 0 & \text{if } \phi \leq 0, \\ 1 & \text{if } \phi > 0. \end{cases} \tag{2.7}$$

Then we have that $\chi^-(\vec{x}) = (1 - H(\phi(\vec{x})))$ and $\chi^+(\vec{x}) = H(\phi(\vec{x}))$. In numerical calculations, the Heaviside function is approximated by a smooth function in order to avoid numerical oddities such as the Dirac function being considered zero everywhere by the discretization. We will return to this later but for now we use this to motivate viewing derivatives of the Heaviside function in the usual sense. By definition the *Dirac delta function* is the directional derivative of the Heaveside function in the outward normal direction, given by

$$\hat{\delta}(\vec{x}) = \nabla H(\phi(\vec{x})) \cdot \vec{N} = H'(\phi(\vec{x}))\nabla\phi(\vec{x}) \cdot \left(\frac{\nabla\phi}{|\nabla\phi|} \right) = H'(\phi(\vec{x}))|\nabla\phi(\vec{x})|. \tag{2.8}$$

In one spatial dimension the delta function is defined as the derivative of the one-dimensional Heaveside function, i.e.

$$\delta(\phi) = H'(\phi).$$

Using this we can rewrite (2.8) as

$$\hat{\delta}(\vec{x}) = \delta(\phi(\vec{x}))|\nabla\phi(\vec{x})|. \tag{2.9}$$

2.1.3 The Level Set Equation

Let us now add dynamics to our interface. Consider a surface S that evolves over time, so that it at time t is given by $S(t)$, where $S(0) = S$. Then for a point $\vec{x}(t) \in S(t)$ we have that it moves according to the ordinary differential equation (ODE)

$$\vec{x}_t(t) = \vec{V}(\vec{x}(t), t). \tag{2.10}$$

Now we wish to find a differential equation for $\phi(\vec{x}, t)$ so that at time t the zero isocontour of $\phi(\vec{x}, t)$ equals $S(t)$. Assuming that the speed function $\vec{V}$ can be extended to $\mathbb{R}^3$ we have the following.

Theorem 2.1 (The level set equation)**.** *Let $\phi(\vec{x}, t)$ be a level set function chosen with the initial condition*

$$\{\vec{x};\ \phi(\vec{x}, t = 0) = 0\} = S.$$

Then, evolving $\phi(t)$ according to

$$\phi_t(\vec{x}, t) + \vec{V}(\vec{x}, t) \cdot \nabla\phi(\vec{x}, t) = 0, \tag{2.11}$$

or

$$\phi_t(\vec{x}, t) + F(\vec{x}, t)|\nabla\phi(\vec{x}, t)| = 0, \tag{2.12}$$

where F is the component of velocity in the normal direction $\vec{N}(\vec{x})$, implies that

$$\{\vec{x};\ \phi(\vec{x}, t) = 0\} = S(t).$$

Proof. We want the zero isocontour of ϕ to equal the propagating surface $S(t)$, so we have the condition

$$\phi(\vec{x}(t), t) = 0, \quad \forall\vec{x}(t) \in S(t).$$

Thus it must hold that for all $\vec{x}(t) \in S(t)$

$$0 = \frac{\partial}{\partial t}\phi(\vec{x}(t), t) = \phi_t + \nabla\phi(\vec{x}(t), t) \cdot \vec{x}_t(t). \tag{2.13}$$

Using (2.10) we get

$$\phi_t + \nabla\phi(\vec{x}(t), t) \cdot \vec{V}(\vec{x}(t), t) = 0, \tag{2.14}$$

and extending this PDE to all points $\vec{x} \in \mathbb{R}^3$ we get (2.11). Now, let $\vec{T}(\vec{x}(t))$ be a tangential surface vector at $\vec{x}(t) \in S(t)$. Then we can write $\vec{V} = V_n\vec{N} + V_t\vec{T}$. By (2.3) we have that $\vec{T} \cdot \nabla\phi = 0$, and furthermore

$$\nabla\phi \cdot \vec{N} = \frac{\nabla\phi}{|\nabla\phi|} \cdot \nabla\phi = |\nabla\phi|.$$

Thus we can rewrite (2.14) as

$$\phi_t(\vec{x}(t),t) + \nabla\phi(\vec{x}(t),t)\cdot(V_n\vec{N} + V_t\vec{T}) = \phi_t(x(\vec{t}),t) + V_n|\nabla\phi(\vec{x}(t),t)| = 0,$$

and extending this PDE to all points $\vec{x} \in \mathbb{R}^3$ we get (2.12). □

Once the level set equation is derived, no information on the topology of the surface is needed. The level set PDE can easily be solved using finite difference schemes and time stepping. Finding the surface's propagation is then only a matter of identifying the zero level set.

2.1.4 Numerical solutions

When solving (2.12) we divide the spatial domain into a three-dimensional grid with grid widths $\Delta x, \Delta y$, and Δz. In our implementation we will consider one grid-point for each voxel, so the grid widths are all set to 1. Now, at time t^n we let $\phi^n_{ijk} = \phi(x_i, y_j, z_k, t^n)$, where $t^{n+1} = t^n + \Delta t$. We will use the explicit forward Euler method for time stepping, which is a first order accurate method. While higher order explicit methods or implicit methods could be used, they are computationally expensive and so we do not explore these here. Let us begin the discussion of discretization by considering the alternative level set equation (2.11),

$$\phi_t(\vec{x},t) = -\vec{V}(\vec{x},t)\cdot\nabla\phi(\vec{x},t). \tag{2.15}$$

Suppose we have computed $\phi(\vec{x},\tilde{t})$ for some $\tilde{t} \geq 0$. Now we wish to determine $\phi(\hat{x},\tilde{t}+t)$, for some $\hat{x} \in \mathbb{R}^3$ and $t > 0$. Locally we can consider the constant vector $\hat{V} = \vec{V}(\hat{x},\tilde{t})$. Setting

$$\phi(\hat{x},\tilde{t}+t) = \phi(\hat{x}-\hat{V}t,\tilde{t}). \tag{2.16}$$

we get a solution to (2.15),

$$\frac{\partial}{\partial t}\phi(\hat{x},\tilde{t}+t) = \frac{\partial}{\partial t}\phi(\hat{x}-\hat{V}t,\tilde{t}) = \frac{\partial}{\partial t}\cdot(\hat{x}-\hat{V}t)\nabla\phi(\hat{x}-\hat{V}t,\tilde{t}) = -\hat{V}\cdot\nabla\phi(\hat{x},\tilde{t}+t).$$

Equation (2.16) tells us that, locally, the value at $\phi(\vec{x},t)$ depends on the initial data located at the offset $-\vec{V}(\vec{x},t)\Delta t$. Now let $\vec{x} = (x,y,z)^T$ and $\vec{V}(\vec{x},t) = (u,v,w)^T$. Consider the forward, backward, and central difference operators given respectively,

$$D^+_x\phi = \frac{\phi_{i+1,j,k} - \phi_{i,j,k}}{\Delta x} \tag{2.17}$$

$$D^-_x\phi = \frac{\phi_{i,j,k} - \phi_{i-1,j,k}}{\Delta x} \tag{2.18}$$

$$D^0_x\phi = \frac{\phi_{i+1,j,k} - \phi_{i-1,j,k}}{2\Delta x}. \tag{2.19}$$

The forward and backward difference are first order accurate, while the central is second order, which can be seen by Taylor expansion. In choosing what difference operator to use for the discretization, an important result is needed.

Theorem 2.2 (The CFL condition)**.** *A numerical approximation of a partial differential equation can converge only if the mathematical domain of dependence is contained in the numerical domain of dependence.*

Proof. See [1] for the original derivation. □

Returning to our discretization of (2.15) we know that the mathematical domain of dependence is locally the values lying along the line $-\vec{V}(\vec{x},t) = (-u,-v,-w)^T$. Thus, if $u > 0$ then information is traveling from left to right along the x-axis and the backward difference should be used, while $u < 0$ implies information is traveling from right to left along the x-axis and a forward difference should be used. This is what is commonly referred to as the *upwinding scheme*. Thus when discretizing the PDE we use

$$\frac{\partial}{\partial x}\phi(\vec{x},t) = \left[\max(0,\vec{V}_1(\vec{x},t))D_x^-\phi_i + \min(0,\vec{V}_1(\vec{x},t))D_x^+\phi_i\right] + \mathcal{O}(\Delta x), \quad (2.20)$$

for the spatial derivatives (equivalently for y, and z partial derivatives). The upwinding scheme can be shown to be stable (see [7] 7.1.1.1 and [18] 3.2) when

$$\Delta t \max\left\{\frac{|\vec{V}_1|}{\Delta x} + \frac{|\vec{V}_2|}{\Delta y} + \frac{|\vec{V}_3|}{\Delta z}\right\} < 1. \quad (2.21)$$

Let us now consider the level set equation, given by (2.12),

$$\phi_t(\vec{x},t) + F(\vec{x},t)|\nabla\phi(\vec{x},t)| = 0, \quad (2.22)$$

This will be the form of the level set equations we derive, and the obvious problem here is how to determine the partial derivatives and equivalent step size restriction. This can be done using the so-called Godunov scheme, see [18], Section 6.2. Here we choose the spatial derivative in the following way when computing $|\nabla\phi(\vec{x},t)|$,

$$\frac{\partial}{\partial x}\phi(\vec{x},t)^2 \approx \begin{cases} \max\left(\max(D_x^-\phi_i,0)^2, \min(D_x^+\phi_i,0)^2\right), & \text{if } F(\vec{x},t) \geq 0, \\ \max\left(\min(D_x^-\phi_i,0)^2, \max(D_x^+\phi_i,0)^2\right), & \text{if } F(\vec{x},t) < 0. \end{cases} \quad (2.23)$$

The time step restriction for stability is then given by (see [13], Section 6.4.1),

$$\Delta t \leq \frac{\min(\Delta x, \Delta y, \Delta z)}{\max|F(\vec{x},t)|}. \quad (2.24)$$

Lastly, when curvature is introduced to regularize the level set we use central differencing as parabolic equations have a domain of dependency in all spatial directions. The curvature can be expanded using the chain rule as follows,

$$\begin{aligned}\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) =& \big(\phi_x^2\phi_{yy} - 2\phi_x\phi_y\phi_{xy} + \phi_y^2\phi_{xx} + \phi_x^2\phi_{zz} - 2\phi_x\phi_z\phi_{xz} + \phi_z^2\phi_{xx} \\ &+ \phi_y^2\phi_{zz} - 2\phi_y\phi_z\phi_{yz} + \phi_z^2\phi_{yy}\big)/|\nabla\phi|^3.\end{aligned} \quad (2.25)$$

To discretize the second partial derivatives finite differencing is applied twice, e.g., $\phi_{xx} \approx D_x^+D_x^-\phi$, and $\phi_{xy} \approx D_x^0D_y^0\phi$, which produces second order accurate approximations.

2.1.5 Signed Distance Function

A distance function $d(\vec{x})$ to a set U is defined by

$$d(\vec{x}) = \min|\vec{x} - \vec{x}_C|, \quad \text{for all } \vec{x}_C \in U.$$

Note that $d(\vec{x}) = 0$ for all $\vec{x} \in \partial U$. A signed distance function is then a function ϕ satisfying $|\phi(\vec{x})| = d(\vec{x})$ for all $\vec{x}$. Recalling the notation for the exterior region, Ω^+, and the interior, Ω^-, we can construct a level set function that is a signed distance function if we set

$$\phi(\vec{x}) = \begin{cases} 0, & \text{if } \vec{x} \in \partial\Omega^-, \\ -d(\vec{x}) & \text{if } \vec{x} \in \Omega^-, \\ d(\vec{x}) & \text{if } \vec{x} \in \Omega^+. \end{cases}$$

So ϕ is a distance function to $\partial\Omega^-$ save for interior points where the distance is taken to be negative. Since $d(\vec{x})$ is the Euclidean distance we have that

$$|\nabla\phi| = 1,$$

for all points that are not equidistant to at least two points on the boundary. While part of the reason for choosing ϕ to be a signed distance function is to create a function that will not cause numerical issues with extreme gradient values, the secondary objective is to have a function where we can only update points on the interface, and then use that ϕ is a signed distance function to update the points close to the interface. This is what is commonly referred to as the *sparse field method*. Updating the entire computational domain requires $\mathcal{O}(N^3)$ operations, where N is the number of grid points in each dimension. Working only in a neighborhood of the zero level set reduces this significantly. If the interface has roughly $\mathcal{O}(N^2)$ points, and we update the level set function only for neighboring grid cells (i.e., a bandwidth of one) the computational complexity is reduced to $\mathcal{O}(N^2)$.

2.1.6 The Euler–Lagrange Equation

Most level set equations are derived from integrals representing an energy that is to be minimized. The method to derive the corresponding PDE is to compute the so called Euler–Lagrange equation. We begin by considering some open set $U \subset \mathbb{R}^n$. Now suppose we have a smooth function $L : \mathbb{R}^n \times \mathbb{R} \times \bar{U} \to \mathbb{R}$. This function is called the *Lagrangian*. To simplify notation in the derivation we will write

$$L = L(p, z, \vec{x}) = L(p_1, \ldots, p_n, z, x_1, \ldots, x_n),$$

for $p \in \mathbb{R}^n, z \in \mathbb{R}, \vec{x} \in U$. The variables p and z will be replaced by ∇u and u, respectively, below. The reason for this placeholder notation is to be able to express derivatives of the functional with respect to partial derivatives of u in a simpler manner. Let us now assume $I[u]$ is of the form

$$I[u] = \int_U L(\nabla u(\vec{x}), u(\vec{x}), \vec{x}) d\vec{x},$$

for functions $u : \bar{U} \to \mathbb{R}$ satisfying some boundary condition. This can be taken to either be a Dirichlet condition,

$$u(\vec{x}) = g(\vec{x}), \quad \text{on } \partial U$$

or a Neumann condition,

$$\nabla u(\vec{x}) \cdot \hat{n}(\vec{x}) = g(\vec{x}), \quad \text{on } \partial U,$$

where $\hat{n}$ is the outward facing unit normal for ∂U. Suppose now that we that a certain function u, satisfying the chosen boundary condition, minimizes $I[\cdot]$. We now construct the PDE that this function solves by taking an arbitrary test function v with compact support (i.e., support in a closed bounded subset of U). Then consider

$$i(\tau) = I[u + \tau v], \quad \tau \in \mathbb{R}.$$

Suppose the Dirichlet condition is chosen. Then, since v has compact support, it holds that $u + \tau v = u = g$ on ∂U. If instead a Neumann condition is chosen, it holds that v having compact support implies it is zero in an open subset containing ∂U and thus $\nabla v = 0$ on ∂U, so again $u + \tau v$ satisfies the boundary condition. Since u is a minimizer of $I[\cdot]$, it holds that

$$i'(0) = 0. \tag{2.26}$$

Now let us explicitly compute this derivative. First observe that

$$i(\tau) = \int_U L(\nabla u + \tau \nabla v, u + \tau v, \vec{x}) d\vec{x}.$$

Thus,

$$i'(\tau) = \int_U \left[\sum_{i=1}^{n} L_{p_i}(\nabla u + \tau \nabla v, u + \tau v, \vec{x}) v_{x_i} + L_z(\nabla u + \tau \nabla v, u + \tau v, \vec{x}) v \right] d\vec{x}.$$

Setting $\tau = 0$ we get from (2.26)

$$0 = i'(0) = \int_U \left[\sum_{i=1}^{n} L_{p_i}(\nabla u, u, \vec{x}) v_{x_i} + L_z(\nabla u, u, \vec{x}) v \right] d\vec{x}. \tag{2.27}$$

Letting $\hat{n} = (n_1, \ldots, n_n)^T$ we can integrate by parts to get

$$\begin{aligned} \int_U L_{p_i}(\nabla u, u, \vec{x}) v_{x_i} &= -\int_U (L_{p_i}(\nabla u, u, \vec{x}))_{x_i} v + \int_{\partial U} L_{p_i}(\nabla u, u, \vec{x}) v \, n_i dS \\ &= -\int_U (L_{p_i}(\nabla u, u, \vec{x}))_{x_i} v. \end{aligned}$$

Thus (2.27) is equivalent to

$$0 = \int_U \left[-\sum_{i=1}^{n} (L_{p_i}(\nabla u, u, \vec{x})_{x_i} + L_z(\nabla u, u, x) \right] v d\vec{x}. \tag{2.28}$$

Since (2.28) holds for all test functions v it must hold that

$$-\sum_{i=1}^{n} (L_{p_i}(\nabla u, u, \vec{x})_{x_i} + L_z(\nabla u, u, x)) = 0, \quad \text{in } U.$$

Dropping the placeholder notation we arrive at the Euler–Lagrange equation,

$$-\sum_{i=1}^{n}\frac{\partial}{\partial x_i}\left(\frac{\partial}{\partial u_{x_i}}L(\nabla u,u,\vec{x})\right)+\frac{\partial}{\partial u}L(\nabla u,u,\vec{x})=0,\quad \text{in } U. \tag{2.29}$$

As an example let us consider the Lagrangian

$$L(\nabla u(\vec{x}),u(\vec{x}),\vec{x})=\frac{1}{2}|\nabla u(\vec{x})|^2.$$

To compute the Euler–Lagrange equation we compute the derivatives,

$$\begin{aligned}\frac{\partial}{\partial u}\frac{1}{2}|\nabla u(\vec{x})|^2 &= 0,\\ \frac{\partial}{\partial u_{x_i}}\frac{1}{2}|\nabla u(\vec{x})|^2 &= u_{x_i},\\ \frac{\partial}{\partial x_i}\left(\frac{\partial}{\partial u_{x_i}}\frac{1}{2}|\nabla u(\vec{x})|^2\right) &= u_{x_ix_i}.\end{aligned}$$

Thus a minimizer of

$$\frac{1}{2}\int_U|\nabla u(\vec{x})|^2d\vec{x},$$

satisfies the PDE

$$\Delta u(\vec{x})=0,\quad \text{in } U.$$

This is known as the *Dirichlet principle*. If we want to perform gradient descent on the Euler–Lagrange equation we can introduce artificial time to get $u(\vec{x},t)$. Then we perform time stepping on

$$u_t(\vec{x},t)=\sum_{i=1}^{n}\frac{\partial}{\partial x_i}\left(\frac{\partial}{\partial u_{x_i}}L(\nabla u(\vec{x},t),u(\vec{x},t),\vec{x})\right)-\frac{\partial}{\partial u}L(\nabla u(\vec{x},t),u(\vec{x},t),\vec{x}). \tag{2.30}$$

To see why this holds we begin by labeling the left hand side of the Euler–Lagrange equation (2.29) as $A[u]$. Then we can write (2.30) as

$$u_t(\vec{x},t)=-A[u].$$

Assume now that u is not a local minimizer of $I[\cdot]$. Then for a fixed v we have

$$i(\tau)=i(0)+\tau i'(0)+\mathcal{O}(\tau^2).$$

We now wish to find v such that moving u in the direction of v (i.e., $u+\tau v$) gives the greatest change to $I[u]$, i.e., we want to find v that maximizes $|i'(0)|$. The right hand side of (2.28) is precisely $i'(0)$,

$$i'(0)=\int_U A[u]vd\vec{x}.$$

This is in fact the inner product of $A[u]$ and v in L^2 space. By the Cauchy-Schwarz inequality we have

$$|i'(0)|=|\langle A[u],v\rangle|\leq\|A[u]\|\|v\|,$$

with equality when $A[u]$ and v are collinear. Thus, choosing $v = A[u]$ gives us the greatest change in $I[u]$ when moving u in its direction. In this case

$$i'(0) = \langle A[u], A[u] \rangle \geq 0,$$

and so to moving u in the direction $-A[u]$ corresponds to the greatest descent in $I[u]$.

2.2 Expectation Maximization

The Expectation Maximization (EM) algorithm, introduced in [3], is an iterative procedure for computing the Maximum Likelihood (ML) estimate for a probabilistic model with unobserved, or missing, data. In general, we wish to find parameters for the model that maximize the likelihood of the observed data. As an example consider a set of observations where each observation comes from one of two Gaussian distributions. However, we do not know which Gaussian a given sample comes from, nor the respective Gaussians' parameters or the likelihood of the sample coming from a particular Gaussian. The EM-algorithm computes the parameters that most likely produced the output. Here the missing data is which Gaussian a given sample comes from. If we knew this computing the parameters would be trivial. In this section we begin by recalling some basic results regarding convex functions to use in deriving the general EM algorithm, following the very intuitive method presented in [24]. We then see how this is applied for models consisting of a linear combination of Gaussians, so-called *Gaussian mixture models*, and finally how K-means clustering can be used to create an initial guess for the parameters.

2.2.1 Convex Functions

We begin by recalling some properties of convex functions. We will skip the proofs as this should be familiar from a calculus course. The proofs can however be found in [24].

Definition 2.3. A real valued function f defined on $I = [a, b]$ is said to be *convex* on I if for all $x_1, x_2 \in I$ and $\lambda \in [0, 1]$, it holds that

$$f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda)f(x_2).$$

If the inequality is strict, we say that f is *strictly convex*. Furthermore, we say that f is *(strictly) concave* if $-f$ is (strictly) convex.

For differentiable functions convexity can be determined by computing the sign of the second derivative across the interval.

Lemma 2.4. *If $f(x) \in C^2(I)$ and $f''(x) \geq 0$ on I, then f is convex on I. If the inequality is strict then f is strictly convex.*

Then it directly follows that

Lemma 2.5. *$-\ln(x)$ is strictly convex on $(0, \infty)$.*

The following inequality is useful in moving summation out of the natural logarithm.

Lemma 2.6 (Jensen's inequality)**.** *If f is convex on I, $x_1, \ldots, x_n \in I$, and $\lambda_1, \ldots, \lambda_n \geq 0$ with $\sum_{i=1}^{n} \lambda_i = 1$, it holds that*

$$f\left(\sum_{i=1}^{n} \lambda_i x_i\right) \leq \sum_{i=1}^{n} \lambda_i f(x_i).$$

Applying this to $-\ln(x)$ which is convex we get

$$\ln \sum_{i=1}^{n} \lambda_i x_i \geq \sum_{i=1}^{n} \lambda_i \ln(x_i). \tag{2.31}$$

2.2.2 The General Algorithm

Each iteration of the EM algorithm consists of two steps. The first is the E-step, where the unobserved data is estimated given the observed data and current parameters estimate. In the M-step, the likelihood function is maximized under the assumption that the unobserved data is known. In the general case it is possible for the EM algorithm not to converge. It is, however, proved in [4] that for the exponential family, e.g., Gaussian distributions, the algorithm converges to a stationary point that is a local maximum or in unusual cases, a saddle point. To avoid the latter, several starting points in the parameter space can be tested as the particular stationary point the algorithm converges to depends on the starting values of the parameters.

We begin the derivation by considering a vector of independently drawn observations $\boldsymbol{X} = (x_1, \ldots, x_N)^t$, where each observation comes from some parametrized family. We wish to find θ that maximizes $p(\boldsymbol{X}|\theta)$, i.e., the ML estimate of θ. To do this we introduce the log likelihood function

$$L(\theta) = \ln p(\boldsymbol{X}|\theta).$$

Since the logarithmic function is strictly increasing, the θ that maximizes $L(\theta)$ will also maximize $p(\boldsymbol{X}|\theta)$. Suppose we have an estimate for θ, denoted θ_n, after the n^{th} iteration of the EM algorithm. The goal is to find a θ that increases the likelihood function, so $L(\theta) > L(\theta_n)$. Using the definition of L we can express this problem as finding θ that maximizes

$$L(\theta) - L(\theta_n) = \ln p(\boldsymbol{X}|\theta) - \ln p(\boldsymbol{X}|\theta_n). \tag{2.32}$$

Let us now introduce the unobserved, or missing variables. It is assumed that observing these would make the maximization of L easier. Denote a realization of the unobserved data as the vector $\boldsymbol{Z}$. The probability $p(\boldsymbol{X}|\theta)$ can then be expressed using the law of total probability as

$$p(\boldsymbol{X}|\theta) = \sum_{\boldsymbol{Z}} p(\boldsymbol{X}|\boldsymbol{Z}, \theta) p(\boldsymbol{Z}|\theta),$$

where we sum over the sample space for $\boldsymbol{Z}$. Using this to rewrite equation (2.32) we get

$$L(\theta) - L(\theta_n) = \ln \sum_{\boldsymbol{Z}} p(\boldsymbol{X}|\boldsymbol{Z}, \theta) p(\boldsymbol{Z}|\theta) - \ln p(\boldsymbol{X}|\theta_n). \tag{2.33}$$

Now, recall the expression derived for the logarithm of a sum (2.31),

$$\ln \sum_{i=1}^{n} \lambda_i x_i \geq \sum_{i=1}^{n} \lambda_i \ln(x_i),$$

with the condition that $\lambda_i \geq 0$, and $\sum_{i=1}^{n} \lambda_i = 1$. This inequality can be used for constants of the form $p(\boldsymbol{Z}|\boldsymbol{X}, \theta_n)$, since it satisfies positivity and summation over $\boldsymbol{Z}$ equals one, as it is a probability measure. We use this to rewrite (2.33) as

$$\begin{aligned}
L(\theta) - L(\theta_n) &= \ln \sum_{\boldsymbol{Z}} p(\boldsymbol{X}|\boldsymbol{Z}, \theta) p(\boldsymbol{Z}|\theta) - \ln p(\boldsymbol{X}|\theta_n) \\
&= \ln \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X}, \theta_n) \left(\frac{p(\boldsymbol{X}|\boldsymbol{Z}, \theta) p(\boldsymbol{Z}|\theta)}{p(\boldsymbol{Z}|\boldsymbol{X}, \theta_n)} \right) - \ln p(\boldsymbol{X}|\theta_n) \cdot 1 \\
&\geq \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X}, \theta_n) \ln \left(\frac{p(\boldsymbol{X}|\boldsymbol{Z}, \theta) p(\boldsymbol{Z}|\theta)}{p(\boldsymbol{Z}|\boldsymbol{X}, \theta_n)} \right) - \ln p(\boldsymbol{X}|\theta_n) \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X}, \theta_n) \\
&= \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X}, \theta_n) \ln \left(\frac{p(\boldsymbol{X}|\boldsymbol{Z}, \theta) p(\boldsymbol{Z}|\theta)}{p(\boldsymbol{Z}|\boldsymbol{X}, \theta_n) p(\boldsymbol{X}|\theta_n)} \right) \\
&=: \Delta(\theta|\theta_n),
\end{aligned}$$

which implies that

$$L(\theta) \geq L(\theta_n) + \Delta(\theta|\theta_n) =: l(\theta|\theta_n). \tag{2.34}$$

Now observe that

$$\begin{aligned}
l(\theta_n|\theta_n) &= L(\theta_n) + \Delta(\theta_n|\theta_n) \\
&= L(\theta_n) + \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X}, \theta_n) \ln \left(\frac{p(\boldsymbol{X}|\boldsymbol{Z}, \theta_n) p(\boldsymbol{Z}|\theta_n)}{p(\boldsymbol{Z}|\boldsymbol{X}, \theta_n) p(\boldsymbol{X}|\theta_n)} \right) \\
&= L(\theta_n) + \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X}, \theta_n) \ln \left(\frac{\frac{p(\boldsymbol{X}, \boldsymbol{Z}|\theta_n)}{p(\boldsymbol{Z}|\theta_n)} p(\boldsymbol{Z}|\theta_n)}{\frac{p(\boldsymbol{X}, \boldsymbol{Z}|\theta_n)}{p(\boldsymbol{X}|\theta_n)} p(\boldsymbol{X}|\theta_n)} \right) \\
&= L(\theta_n) + \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X}, \theta_n) \ln 1 \\
&= L(\theta_n).
\end{aligned}$$

Together with (2.34) we have that

$$\begin{cases} L(\theta) \geq l(\theta|\theta_n), \\ L(\theta_n) = l(\theta_n|\theta_n). \end{cases}$$

Thus any θ that increases $l(\theta|\theta_n)$ must also increase the value of $L(\theta)$. Therefore

θ_{n+1} should be chosen as

$$\begin{aligned}
\theta_{n+1} &= \arg\max_{\theta}\{l(\theta|\theta_n)\} \\
&= \arg\max_{\theta}\left\{L(\theta_n) + \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X},\theta_n)\ln\left(\frac{p(\boldsymbol{X}|\boldsymbol{Z},\theta)p(\boldsymbol{Z}|\theta)}{p(\boldsymbol{Z}|\boldsymbol{X},\theta_n)p(\boldsymbol{X}|\theta_n)}\right)\right\} \\
&= \arg\max_{\theta}\left\{\sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X},\theta_n)\ln\left(p(\boldsymbol{X}|\boldsymbol{Z},\theta)p(\boldsymbol{Z}|\theta)\right)\right. \\
&\quad \left. + L(\theta_n) - \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X},\theta_n)\ln\left(p(\boldsymbol{Z}|\boldsymbol{X},\theta_n)p(\boldsymbol{X}|\theta_n)\right)\right\} \\
&= \arg\max_{\theta}\left\{\sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X},\theta_n)\ln\left(\frac{p(\boldsymbol{X},\boldsymbol{Z},\theta)}{p(\boldsymbol{Z},\theta)}\frac{p(\boldsymbol{Z},\theta)}{p(\theta)}\right)\right\} \\
&= \arg\max_{\theta}\left\{\sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X},\theta_n)\ln p(\boldsymbol{X},\boldsymbol{Z}|\theta)\right\} \\
&= \arg\max_{\theta}\left\{E_{\boldsymbol{Z}|\boldsymbol{X},\theta_n}(\ln p(\boldsymbol{X},\boldsymbol{Z}|\theta)\right\}.
\end{aligned}$$

Thus the EM algorithm consists of iterating the following:

1. **E-step:** Determine the conditional expectation $E_{\boldsymbol{Z}|\boldsymbol{X},\theta_n}(\ln p(\boldsymbol{X},\boldsymbol{Z}|\theta))$.
2. **M-step:** Maximize $E_{\boldsymbol{Z}|\boldsymbol{X},\theta_n}(\ln p(\boldsymbol{X},\boldsymbol{Z}|\theta))$ with respect to θ to find θ_{n+1}.

2.2.3 EM for Gaussian Mixtures

Let us now see how the EM algorithm can be used to compute the ML estimate of a Gaussian mixture model's (GMM) parameters. We will in large follow the arguments from [19], Chapter 9. A GMM consists of K Gaussians,

$$\mathcal{N}(x|\mu_k,\sigma_k) = \frac{1}{\sqrt{2\pi}\sigma_k}e^{-\frac{(x-\mu_k)^2}{2\sigma_k^2}}, \quad k = 1,\ldots,K,$$

where $\mu_k \in \mathbb{R}$ denotes the mean and $\sigma_k \in \mathbb{R}$ the standard deviation. The model, furthermore, consists of probabilities

$$\begin{aligned}
&\pi_k \geq 0, \quad k = 1,\ldots,K, \\
&\sum_{k=1}^{K}\pi_k = 1,
\end{aligned} \tag{2.35}$$

that indicate the probability of a sample being drawn from Gaussian k. We are given a vector of samples $\boldsymbol{X} = (x_1,\ldots,x_N)^T$ but do not know which distribution each sample was drawn from. Given this, we wish to find the ML estimate of the parameter $\theta = (\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\sigma})^T$, where $\boldsymbol{\pi} = (\pi_1,\ldots,\pi_K)^T$, $\boldsymbol{\mu} = (\mu_1,\ldots,\mu_K)^T$, and $\boldsymbol{\sigma} = (\sigma_1,\ldots,\sigma_K)^T$. We begin by introducing the K-dimensional unobserved random variable $\boldsymbol{z}_n = (z_{n1},\ldots,z_{nK})$ that has a 1-of-K representation, i.e., one element z_k is equal to one, and all others are equal to zero. This is the missing variable that determines which distribution x_n was drawn from. By definition we must then have

$$p(z_{nk} = 1) = \pi_k, \quad k = 1,\ldots,K,$$

since $z_{nk} = 1$ if sample x_n is drawn from Gaussian k, and the probability of this occurring is π_k. We can express the 1-of-K distribution for $\boldsymbol{z}_n$ as

$$p(\boldsymbol{z}_n) = \prod_{k=1}^{K} \pi_k^{z_{nk}}. \tag{2.36}$$

The conditional distribution of the random variable X_n, observed as x_n, given that element k of $\boldsymbol{z}_n$ is equal to one is thus given by

$$p(x|z_{nk} = 1) = \mathcal{N}(x|\mu_k, \sigma_k),$$

which can also be expressed as

$$p(x|\boldsymbol{z}_n) = \prod_{k=1}^{K} \mathcal{N}(x|\mu_k, \sigma_k)^{z_{nk}}. \tag{2.37}$$

The marginal distribution of X_n can then be obtained by summing the joint distribution $p(\boldsymbol{z}_n)p(x|\boldsymbol{z}_n)$ over all the K possible states of $\boldsymbol{z}_n$,

$$p(x) = \sum_{k=1}^{K} p(z_{nk} = 1)p(x|z_{nk} = 1) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \sigma_k). \tag{2.38}$$

This is what is referred to as a *Gaussian mixture*. Here we see how for every realization x_n there is a corresponding K-dimensional missing observation $\boldsymbol{z}_n$. Let us now derive the EM algorithm for this model. We have observed data $\boldsymbol{X}$ and denote the corresponding missing discrete data as a N by K matrix $\boldsymbol{Z}$. Each row $\boldsymbol{z}_n$ is the corresponding missing data for the observation x_n, so element $z_{nk} = 1$ if observation x_n was drawn from Gaussian k. Using (2.36), (2.37), and independence of the samples, we have that the likelihood function has the form

$$p(\boldsymbol{X}, \boldsymbol{Z}|\theta) = p(\boldsymbol{Z}|\boldsymbol{\pi})p(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\mu}, \boldsymbol{\sigma}) = \prod_{n=1}^{N} \prod_{k=1}^{K} \pi_k^{z_{nk}} \mathcal{N}(x_n|\mu_k, \sigma_k)^{z_{nk}}.$$

The log likelihood function is then given by

$$\ln p(\boldsymbol{X}, \boldsymbol{Z}|\theta) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk}[\ln \pi_k + \ln \mathcal{N}(x_n|\mu_k, \sigma_k)].$$

Since the samples are independent, we have that the rows of $\boldsymbol{Z}$, denoted $\boldsymbol{z}_n$, are as well. Thus the expected value of z_{nk} under the posterior distribution depends solely on x_n and is given by

$$\begin{aligned} E_{\boldsymbol{Z}|\boldsymbol{X},\theta_n}(z_{nk}) &= \sum_{z_{nk}} z_{nk} p(z_{nk}|\boldsymbol{X}, \theta_n) = p(z_{nk} = 1|x_n, \theta_n) \\ &= \frac{p(z_{nk} = 1)p(x_n|z_{nk} = 1)}{p(x_n)} = \frac{\pi_k \mathcal{N}(x_n|\mu_k^n, \sigma_k^n)}{\sum_{i=1}^{K} \pi_i \mathcal{N}(x_n|\mu_i^n, \sigma_i^n)} \\ &:= \gamma(z_{nk}), \end{aligned}$$

where $\theta_n = (\boldsymbol{\pi}^n, \boldsymbol{\mu}^n, \boldsymbol{\mu}^n)$ denotes the current estimate of $\theta = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\sigma})$. Thus we have that

$$\begin{aligned} E_{\boldsymbol{Z}|\boldsymbol{X},\theta_n}(\ln p(\boldsymbol{X}, \boldsymbol{Z}|\theta)) &= \sum_{n=1}^{N}\sum_{k=1}^{K} E_{\boldsymbol{Z}|\boldsymbol{X},\theta_n}(z_{nk})[\ln \pi_k + \ln \mathcal{N}(x_n|\mu_k, \sigma_k)] \\ &= \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma(z_{nk})[\ln \pi_k + \ln \mathcal{N}(x_n|\mu_k, \sigma_k)]. \end{aligned}$$

We have that the E step consists of evaluating $\gamma(z_{nk})$ using the current parameter estimation θ_n, followed by maximization of

$$\mathcal{Q}(\theta, \theta_n) := E_{\boldsymbol{Z}|\boldsymbol{X},\theta_n}(\ln p(\boldsymbol{X}, \boldsymbol{Z}|\theta)) = \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma(z_{nk})\big(\ln \pi_k + \ln \mathcal{N}(x_n|\mu_k, \sigma_k)\big),$$

with respect to $\theta = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\sigma})^T$. Using the definition of $\mathcal{N}$,

$$\mathcal{N}(x|\mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

we can write $\mathcal{Q}$ as

$$\mathcal{Q}(\theta, \theta_n) = \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma(z_{nk})\big(\ln \pi_k - \ln(\sqrt{2\pi}\sigma_k) - \frac{(x-\mu_k)^2}{2\sigma_k^2}\big). \tag{2.39}$$

Setting the derivative of $\mathcal{Q}$ with respect to μ_i equal to zero we get

$$0 = \sum_{n=1}^{N} \gamma(z_{ni}) \frac{(x-\mu_i)}{\sigma_k^2},$$

which implies that

$$\mu_i = \frac{1}{N_i} \sum_{n=1}^{N} \gamma(z_{ni}) x_n, \tag{2.40}$$

where we let $N_i := \sum_{n=1}^{N} \gamma(z_{ni})$. The expression for μ_i can be seen as the weighted mean of all the samples, where the weighting factor is the posterior probability that x_n was sampled from Gaussian i. Setting the derivative of (2.39) to zero with respect to σ_i we get

$$0 = \sum_{n=1}^{N} \gamma(z_{ni}) \frac{\big((x_n - \mu_i)^2 - \sigma_i^2\big)}{\sigma_i^3},$$

which implies that

$$\sigma_i = \sqrt{\frac{1}{N_i} \sum_{n=1}^{N} \gamma(z_{ni})(x_n - \mu_i)^2}, \tag{2.41}$$

which is similar to fitting one Gaussian to the data set, but again with each sample weighted by the posterior probability of it belonging to Gaussian i. Lastly

we maximize Q with respect to π_i. This is however a constrained optimization problem since we have that (2.35) must hold. Adding this to the objective function through a Lagrange multiplier we can instead consider maximizing

$$\mathcal{L}(\boldsymbol{\pi}, \lambda) = \sum_{n=1}^{N}\sum_{k=1}^{K}\gamma(z_{nk})\big(\ln \pi_k - \ln(\sqrt{2\pi}\sigma_k) - \frac{(x-\mu_k)^2}{2\sigma_k^2}\big) + \lambda\left(\sum_{k=1}^{K}\pi_k - 1\right).$$

Setting the derivative of $\mathcal{L}$ to zero with respect to π_i we get

$$0 = \sum_{n=1}^{N}\frac{\gamma(z_{ni})}{\pi_i} + \lambda, \tag{2.42}$$

which is equivalent to

$$0 = \sum_{n=1}^{N}\sum_{k=1}^{K}\frac{\pi_k\gamma(z_{ni})}{\pi_i} + \lambda\sum_{k=1}^{K}\pi_k = \sum_{n=1}^{N}1 + \lambda,$$

which gives $\lambda = -N$. Using this in (2.42) we get

$$\pi_i = \frac{N_i}{N} = \frac{1}{N}\sum_{n=1}^{N}\gamma(z_{ni}), \tag{2.43}$$

so the weight π_i is the average responsibility $\gamma(z_{ni})$ given to the i^{th} Gaussian component. Notice that the update formulas do not give a closed form solution to the parameters since $\gamma(z_{nk})$ depends on the parameters. The EM algorithm for a GMM can be summarized as follows.

1. Initialize the means μ_k^0, the standard deviations σ_k^0, and the mixing weights π_k^0. Set $t = 0$.
2. **E-step:** Evaluate the responsibilities using the current estimated parameters
$$\gamma(z_{nk}) = p(z_{nk} = 1|x_n, \theta_n) = \frac{\pi_k\mathcal{N}(x_n|\mu_k^t, \sigma_k^t)}{\sum_{i=1}^{K}\pi_i\mathcal{N}(x_n|\mu_i^t, \sigma_i^t)}.$$
3. **M-step:** Update the estimation of the parameters using
$$\mu_i^{t+1} = \frac{1}{N_i}\sum_{n=1}^{N}\gamma(z_{ni})x_n,$$
$$\sigma_i^{t+1} = \sqrt{\frac{1}{N_i}\sum_{n=1}^{N}\gamma(z_{ni})(x_n - \mu_i)^2},$$
$$\pi_i^{t+1} = \frac{N_i}{N},$$
where
$$N_i = \sum_{n=1}^{N}\gamma(z_{ni}).$$
4. Check for convergence of the parameters or of the log likelihood function
$$\ln p(\boldsymbol{X}|\boldsymbol{\pi^t}, \boldsymbol{\mu^t}, \boldsymbol{\sigma^t}) = \sum_{n=1}^{N}\ln\left(\sum_{k=1}^{K}\pi_k\mathcal{N}(x_n|\mu_k^t, \sigma_k^t)\right).$$
If convergence criterion is not satisfied set $t := t + 1$ and return to step 2.

2.2.4 K-means Clustering

The first step of the EM algorithm is to estimate the parameters. The K-means algorithm finds clusters such that the distance between points in each cluster is small. Suppose we have K fixed as the number of clusters we are considering, and we have a set of data points $\boldsymbol{x} = (x_1, \ldots, x_n)^T$. Then we introduce the vector $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_K)^T$. Here μ_k can be considered a "prototype" for cluster k, or the "center" of the cluster. Next we introduce assignment to a specific cluster through the binary variable $r_{nk} \in \{0, 1\}$, where x_n being assigned to cluster k implies $r_{nk} = 1$ and $r_{nj} = 0$ for all $j \neq k$. The objective function we wish to minimize is then given by

$$E = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} |x_n - \mu_k|^2,$$

which is the sum of the squared distance between each point and the assigned cluster's centre μ_k. We wish to minimize E with respect to r_{nk} and μ_k. We begin with choosing some value for the μ_k, which can for instance be chosen to be spread out evenly across the interval that the data lies within. Then we minimize E with respect to the assignment variables r_{nk} followed by minimization with respect to μ_k. This is similar to the EM algorithm and in fact the K-means algorithm can be seen as a limiting case of the EM algorithm, where the variances are treated as fixed and allowed to tend towards zero. A more detailed discussion of this can be found in [19], Section 9.3.2.

We begin by minimizing E with respect to the r_{nk}. This is trivial since E is a linear function of r_{nk} and the terms involving different n are independent and so we can maximize with respect to each n individually. Thus we assign the data points to the closest cluster centre,

$$r_{nk} = \begin{cases} 1, & \text{if } k = \arg\min_j |x_n - \mu_j|^2, \\ 0, & \text{otherwise.} \end{cases}$$

Next we set the derivative of E with respect to μ_k to zero and get

$$2 \sum_{n=1}^{N} r_{nk}(x_n - \mu_k) = 0,$$

which implies that

$$\mu_k = \frac{\sum_{n=1}^{N} r_{nk} x_n}{\sum_{n=1}^{N} r_{nk}},$$

i.e., the mean of the data points assigned to the cluster. The convergence properties of the K-means algorithm was studied in [2]. Convergence is guaranteed, however it may be to a local optimum rather than global. In summary the K-means algorithm proceeds as follows.

1. Initialize μ_k with some values.
2. Update assignments

$$r_{nk} = \begin{cases} 1, & \text{if } k = \arg\min_j |x_n - \mu_j|^2, \\ 0, & \text{otherwise.} \end{cases}$$

3. Update the cluster means

$$\mu_k = \frac{\sum_{n=1}^{N} r_{nk} x_n}{\sum_{n=1}^{N} r_{nk}}.$$

4. Check for convergence by computing the objective function

$$E = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} |x_n - \mu_k|^2.$$

If convergence criterion is not fulfilled, return to step 2.

Once the algorithm has converged we can use μ_k as an initial estimation of the means in the EM algorithm. The weights and standard deviations are then set to

$$\pi_k = \frac{1}{N} \sum_{n=1}^{N} r_{nk},$$

and

$$\sigma_k = \sqrt{\frac{\sum_{n=1}^{N} r_{nk}(x_n - \mu_k)^2}{\sum_{n=1}^{N} r_{nk}}}.$$

2.3 Image Segmentation

The idea behind active contour models (ACM) is to have a curve evolve according to constraints of a given image I. Let us begin with considering a simpler case in $\mathbb{R}^2$. Let Ω be a bounded open subset of $\mathbb{R}^2$ and $I : \bar{\Omega} \to \mathbb{R}$ a given grayscale image, and $C(s) : [0, 1] \to \mathbb{R}^2$ be a parametrized curve. In the original ACM an edge-detector function g is used. They can generally be defined by a positive and decreasing function such that

$$\lim_{x \to \infty} g(x) = 0.$$

In the geodesic ACM, [11], the edge detector is chosen to be

$$g(|\nabla I|) = \frac{1}{1 + |\nabla G_\sigma * I|^2},$$

where $G_\sigma * I$ is a smoothing of I obtained through convolution with the Gaussian

$$G_\sigma(x, y) = \frac{1}{\sqrt{(2\pi)}\sigma} e^{-\frac{(x^2+y^2)}{2\sigma^2}}.$$

We then wish to find a minimizer to

$$E[C] = \int_0^1 g(|\nabla I(C(s))|)|C'(s)|ds.$$

This energy is not minimized solely by a curve with short length but also one that passes through regions where the image's gradient is large. The issue with models relying on an edge detector function is that in practice discrete gradients are bounded and then g is never zero, and thus the curve may pass through the boundary. Furthermore, if the image noisy then the smoothing must be strong (i.e., choosing σ to be large in G_σ), this will however also smooth the edges. Lastly, such models fail at segmenting objects with a very smooth boundary, a characteristic common in medical images.

2.3.1 The Chan–Vese Method

In the highly influential paper *Active Contours Without Edges*, [15], Chan and Vese derived a *region based* ACM that constructs the energy functional based on image fitting terms rather than image gradients. We will present the model in three dimensions since we are interested in segmenting volumes, however the two dimensional case follows the same arguments.

To begin, suppose we have a surface S in $\Omega \subset \mathbb{R}^3$ that is the boundary of an open subset $\Omega^- \subset \Omega$, so $S = \partial\Omega^-$. The exterior is then given by $\Omega^+ = \Omega \setminus \overline{\Omega^-}$. To understand the basic idea of the model consider an image I consisting of two regions, each with approximately constant intensity of distinct values, I^-, and I^+. The object to be segmented is represented by the region with intensity I^-. Denote its boundary by S_0. Then we have $I \approx I^-$ in the interior Ω^-, and $I \approx I^+$ in the exterior Ω^+. Now consider

$$F^-(S) + F^+(S) = \int_{\Omega^-} |I(\vec{x}) - c_1|^2 d\vec{x} + \int_{\Omega^+} |I(\vec{x}) - c_2|^2 d\vec{x}, \tag{2.44}$$

where S is any other surface, and the constants c_1, c_2 depending on S, are the averages of I inside and outside S, Ω^- and Ω^+, respectively. In this case it is obvious that S_0, the boundary of the object, minimizes

$$\inf_S \{F^-(S) + F^+(S)\} \approx 0 \approx F^-(S_0) + F^+(S_0).$$

In the Chan–Vese model this "fitting" term is minimized together with a regularization term for the area of the surface and volume of the object. As earlier we embed the surface as the zero level set of a level set function $\phi(\vec{x})$. Then we have as in Section 2.1.1

$$\begin{cases} S = \partial\Omega^- = \{\vec{x} \in \Omega; \phi(\vec{x}) = 0\}, \\ \Omega^- = \{\vec{x} \in \Omega; \phi(\vec{x}) < 0\}, \\ \Omega^+ = \{\vec{x} \in \Omega; \phi(\vec{x}) > 0\}. \end{cases}$$

It is shown in [5] (section 6.1) that

$$\int_{\mathbb{R}^3} f(\vec{x})\delta(g(\vec{x}))d\vec{x} = \int_{g^{-1}(0)} \frac{f(\vec{x})}{|\nabla g(\vec{x})|} dS.$$

Letting $g(\vec{x}) = \phi(\vec{x})$ and $f(\vec{x}) = |\nabla\phi(\vec{x})|$ we get a formula for computing the area of the surface,

$$A(S) = \int_{\partial\Omega^-} dS = \int_\Omega \delta(\phi(\vec{x}))|\nabla\phi(\vec{x})|d\vec{x}. \tag{2.45}$$

The volume of Ω^-, the interior, is computed as

$$V(S) = \int_{\Omega-} d\vec{x} = \int_{\phi<0} d\vec{x} = \int_\Omega [1 - H(\phi(\vec{x}))]d\vec{x}.$$

The fitting energies (2.44) are given by

$$\begin{aligned} F^-(S) &= \int_{\Omega^-} |I(\vec{x}) - c_1|^2 d\vec{x} = \int_{\phi<0} |I(\vec{x}) - c_1|^2 d\vec{x} \\ &= \int_\Omega |I(\vec{x}) - c_1|^2 [1 - H(\phi(\vec{x}))]d\vec{x}, \end{aligned}$$

and likewise,

$$F^+(S) = \int_{\Omega^+} |I(\vec{x}) - c_2|^2 d\vec{x} = \int_{\phi>0} |I(\vec{x}) - c_2|^2 d\vec{x}$$
$$= \int_{\Omega} |I(\vec{x}) - c_2|^2 H(\phi(\vec{x})) d\vec{x}.$$

Then, the energy functional $E_{\text{cv}}(\phi, c1, c2)$ can be expressed as

$$\begin{aligned} E_{\text{cv}}(\phi, c1, c2) &= \lambda_1 F^-(S) + \lambda_2 F^+(S) + \mu A(S) + \nu V(S) \\ &= \lambda_1 \int_{\Omega} |I(\vec{x}) - c_1|^2 [1 - H(\phi(\vec{x}))] d\vec{x} \\ &+ \lambda_2 \int_{\Omega} |I(\vec{x}) - c_2|^2 H(\phi(\vec{x})) d\vec{x} \\ &+ \mu \int_{\Omega} \delta(\phi(\vec{x})) |\nabla \phi(\vec{x})| d\vec{x} \\ &+ \nu \int_{\Omega} [1 - H(\phi(\vec{x}))] d\vec{x}, \end{aligned} \tag{2.46}$$

where $\lambda_1, \lambda_2, \mu, \nu$ are positive parameters. The energy E_{cv} can be minimized with respect to c_1, to get

$$c_1(\phi) = \frac{\int_{\Omega} I(\vec{x})[1 - H(\phi(\vec{x}))] d\vec{x}}{\int_{\Omega} [1 - H(\phi(\vec{x}))] d\vec{x}}, \tag{2.47}$$

the average intensity of the interior, as one might expect. Similarly minimizing E_{cv} with respect to c_2 gives

$$c_2(\phi) = \frac{\int_{\Omega} I(\vec{x}) H(\phi(\vec{x})) d\vec{x}}{\int_{\Omega} H(\phi(\vec{x})) d\vec{x}}, \tag{2.48}$$

In order to compute the Euler–Lagrange equation of E_{cv} we consider regularized approximations to H and δ, denoted H_ϵ and δ_ϵ, respectively. Choosing

$$H_\epsilon(x) = \frac{1}{2}\left(1 + \frac{2}{\pi} \arctan\left(\frac{x}{\epsilon}\right)\right),$$

then

$$\delta_\epsilon(x) = H'_\epsilon(x) = \frac{1}{\pi} \cdot \frac{\epsilon}{\epsilon^2 + x^2}.$$

These regularizations converge to H and δ as $\epsilon \to 0$. Let us now compute the associated Euler–Lagrange equation (2.29) to the regularized functional E^ϵ_{cv}. The Lagrangian is given by

$$\begin{aligned} L(\nabla\phi, \phi, \vec{x}) =& \lambda_1 |I(\vec{x}) - c_1|^2 \big(1 - H_\epsilon(\phi(\vec{x}))\big) + \lambda_2 |I(\vec{x}) - c_2|^2 H_\epsilon(\phi(\vec{x})) \\ &+ \mu \delta_\epsilon(\phi(\vec{x})) |\nabla\phi(\vec{x})| + \nu\big(1 - H_\epsilon(\phi(\vec{x}))\big). \end{aligned}$$

Letting $\vec{x} = (x_1, x_2, x_3)^T$, we get

$$\frac{\partial}{\partial \phi_{x_i}} L = \frac{\partial}{\partial \phi_{x_i}} \mu \delta_\epsilon(\phi) |\nabla\phi| = \mu \delta_\epsilon(\phi) \frac{\phi_{x_i}}{|\nabla\phi|},$$

$$\frac{\partial}{\partial x_i}\left(\frac{\partial}{\partial \phi_{x_i}} L\right) = \frac{\partial}{\partial x_i}\left(\mu \delta_\epsilon(\phi) \frac{\phi_{x_i}}{|\nabla\phi|}\right) = \mu \delta'_\epsilon(\phi) \frac{\phi^2_{x_i}}{|\nabla\phi|} + \mu \delta_\epsilon(\phi) \frac{\partial}{\partial x_i}\left(\frac{\phi_{x_i}}{|\nabla\phi|}\right),$$

$$\frac{\partial}{\partial \phi} L = -\lambda_1 (I(\vec{x}) - c_1)^2 \delta_\epsilon(\phi) + \lambda_2 (I(\vec{x}) - c_2)^2 \delta_\epsilon(\phi) + \mu \delta'_\epsilon(\phi) |\nabla\phi| - \nu \delta_\epsilon(\phi).$$

From (2.29) we get

$$\begin{aligned}
0 &= -\sum_{i=1}^{3} \frac{\partial}{\partial x_i}\left(\frac{\partial}{\partial \phi_{x_i}} L(\nabla\phi, \phi, \vec{x})\right) + \frac{\partial}{\partial \phi} L(\nabla\phi, \phi, \vec{x}) \\
&= -\sum_{i=1}^{3} \mu\delta'_\epsilon(\phi)\frac{\phi_{x_i}^2}{|\nabla\phi|} + \mu\delta_\epsilon(\phi)\frac{\partial}{\partial x_i}\left(\frac{\phi_{x_i}}{|\nabla\phi|}\right) \\
&+ \big(-\lambda_1(I(\vec{x}) - c_1)^2\delta_\epsilon(\phi) + \lambda_2(I(\vec{x}) - c_2)^2\delta_\epsilon(\phi) + \mu\delta'_\epsilon(\phi)|\nabla\phi| - \nu\delta_\epsilon(\phi)\big) \\
&= -\mu\delta'_\epsilon(\phi)|\nabla\phi| - \mu\delta_\epsilon(\phi)\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) \\
&+ \big(-\lambda_1(I(\vec{x}) - c_1)^2\delta_\epsilon(\phi) + \lambda_2(I(\vec{x}) - c_2)^2\delta_\epsilon(\phi) + \mu\delta'_\epsilon(\phi)|\nabla\phi| - \nu\delta_\epsilon(\phi)\big) \\
&= \delta_\epsilon(\phi)\left[-\lambda_1(I(\vec{x}) - c_1)^2 + \lambda_2(I(\vec{x}) - c_2)^2 - \mu\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) - \nu\right].
\end{aligned}$$

This yields the gradient descent equation (2.30),

$$\phi_t(\vec{x}, t) = \delta_\epsilon(\phi)\left[\lambda_1(I(\vec{x}) - c_1)^2 - \lambda_2(I(\vec{x}) - c_2)^2 + \mu\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) + \nu\right] \tag{2.49}$$

with the initial condition being the level set function ϕ_0 representing the initial surface S_0 as its zero level set, so $\phi(\vec{x}, 0) = \phi_0(\vec{x})$. Since the tracked interface is to stay within Ω we use the Neumann condition,

$$\nabla\phi(\vec{x})\cdot\hat{n}(\vec{x}) = 0, \quad \vec{x} \in \partial\Omega, \tag{2.50}$$

where $\hat{n}$ is the outward facing unit normal for ∂U. Thus if the zero level set is located at the boundary and the force term pushes the surface along the outward facing normal, $|\nabla\phi|$ will be zero when evaluated according to (2.20), thus only allowing the surface to stay stationary or contract at points located at the boundary. At this point it should be noted that the descent equation (2.49) is not a level set equation of the form (2.12) since we have δ_ϵ instead of $|\nabla\phi|$. In [10] it is motivated that $\delta_\epsilon(\phi)$ can be replaced for $|\nabla\phi|$ since this amounts to time rescaling, and does not affect the steady state solution while simultaneously removing stiffness near the level sets. Since both factors are positive the direction of descent is not affected. This gives us the level set equation,

$$\begin{aligned}
\phi_t(\vec{x}, t) &= \Big[\lambda_1(I(\vec{x}) - c_1)^2 - \lambda_2(I(\vec{x}) - c_2)^2 \\
&\qquad + \mu\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) + \nu\Big]|\nabla\phi|, \quad \text{in } (0,\infty)\times\Omega, \\
\phi(\vec{x}, 0) &= \phi_0(\vec{x}), \quad \text{in } \Omega, \\
\frac{\partial\phi}{\partial\hat{n}} &= 0, \quad \text{on } \partial\Omega.
\end{aligned} \tag{2.51}$$

2.3.2 The Threshold Method

In the Chan–Vese descent equation (2.51), we note that the image driven force is positive or negative, depending on how close the voxel's intensity is the respective mean intensities for the exterior and interior. In the threshold method

an image driven force is used to cause the contour to expand outward only when the image's intensity at a given point is within a given threshold. In [17] a level set equation for image segmentation is introduced of the form

$$\phi_t(\vec{x},t) = \left[\alpha\big(|I(\vec{x}) - T| - \epsilon\big) + (1-\alpha)\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right)\right]|\nabla\phi(\vec{x},t)|. \tag{2.52}$$

Here $\alpha \in [0,1]$ controls how much the external force should be factored in compared to the internal smoothing force from the curvature. The parameter T controls the intensity of the interior region that we are looking to segment, and ϵ controls the range of intensities that should be considered inside. Thus, when $I(\vec{x}) \in (T-\epsilon, T+\epsilon)$ the internal force is negative causing the surface to expand, and when $I(\vec{x})$ falls outside this region, the force becomes positive causing the surface to contract. These parameters are decided at the start of evolution and are not updated.

The existing level set method we will be comparing the new proposed method against uses this level set function to control propagation. To decide the parameters and initialize the level set function the user is asked to draw a line across the tumor, illustrated in Figure 2.1. The level set function is initialized as the signed distance to a sphere located at the center of the drawn line, and the parameters estimated from the distribution along the line for the interior, and along an extrusion of the line for the exterior.

Figure 2.1: A line is drawn across the region that should be segmented.

Let us end with noting how an energy functional would be formulated in order for the variational approach to give rise to the level set equation (2.52). It is easily seen from studying the derivation of the level set equation for the Chan–Vese energy functional (2.46) in the previous section, that the energy

functional

$$E_{\text{thresh}}(\phi) = \alpha\Big(\int_{\Omega} |I(\vec{x}) - T|[1 - H(\phi(\vec{x}))]d\vec{x} + \int_{\Omega} \epsilon\, H(\phi(\vec{x}))d\vec{x}\Big) + (1-\alpha)\int_{\Omega} \delta(\phi(\vec{x}))|\nabla\phi(\vec{x})|d\vec{x},$$

gives rise to the level set function (2.52). With this formulation we see that the energy of the interior intensities is their distance from T, while the energy of exterior intensities is set to ϵ regardless of their values. Therefore, in the simple case with two regions where one has intensities between $T - \epsilon$ and $T + \epsilon$, and the second region has intensities outside this interval, the surface with minimal energy is attained at the border of the two regions.

2.3.3 The Proposed Method

The problem with many region based active contour models such as the two thus far presented is the assumption that the region of interest has a single intensity density coupled with noise. In the case of brain tumors with necrotic regions there is clearly more than one intensity distribution, as seen in Figure 2.2. The same of course applies to the exterior region.

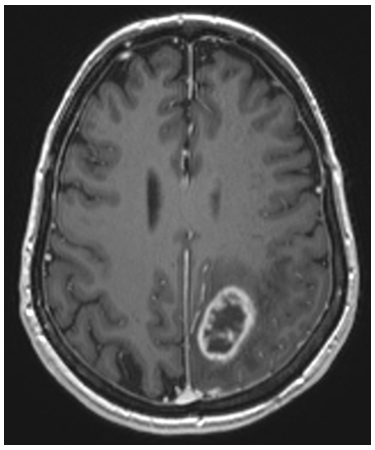


Figure 2.2: The tumor encompasses a necrotic core giving rise to two distinct distributions of intensities.

In this section we propose a new model inspired by the work in [20]. Mainly we will modify the way in which the probabilistic models for the interior and exterior are chosen while also introducing a probability weighting term and a modified initialization procedure. It is observed in [20] that level set segmentation can be expressed as a problem of Bayesian inference, in which the goal is to compute the most likely level set function ϕ given the image $I : \Omega \to \mathbb{R}$. Thus we wish to maximize posterior likelihood function given by

$$p(\phi|I) = \frac{p(I|\phi)p(\phi)}{p(I)} \propto p(I|\phi)p(\phi), \tag{2.53}$$

given equal prior probability of observing a specific intensity. Maximizing the likelihood function is equivalent to minimizing the negative log likelihood function

$$-\ln[p(I|\phi)p(\phi)] = -\ln p(I|\phi) - \ln p(\phi). \tag{2.54}$$

The shape prior term $p(\phi)$ is designed to regulate the surface area and is therefore chosen to be

$$p(\phi) = \exp\left(-(1-\alpha)\int_\Omega \delta(\phi(\vec{x}))|\nabla\phi(\vec{x})|d\vec{x}\right),$$

which implies that

$$-\ln p(\phi) = (1-\alpha)\int_\Omega \delta(\phi(\vec{x}))|\nabla\phi(\vec{x})|d\vec{x}.$$

As for the threshold method we will use the parameter $\alpha \in [0,1]$ to control the degree of smoothing. The posterior image likelihood function is chosen to be

$$p(I|\phi) = \prod_{\vec{x}\in\Omega} \left[p(I(\vec{x})|\phi(\vec{x}))\right]^{\alpha\, d\vec{x}},$$

where the exponent $d\vec{x}$ gives the volume of the cell defined by the grid, which as we will see gives us an appropriate limit as $d\vec{x}$ tends towards zero. Note in this formulation how choosing α to be close to zero implies that $p(I|\phi)$ will stay close to 1, causing this term not to affect maximization greatly. Taking the negative log likelihood we get

$$\begin{aligned} -\ln p(I|\phi) &= \alpha \sum_{\vec{x}\in\Omega} -\ln p(I(\vec{x})|\phi(\vec{x}))d\vec{x} \\ &\to \alpha \int_{\vec{x}\in\Omega} -\ln p(I(\vec{x})|\phi(\vec{x}))d\vec{x}, \quad \text{as } d\vec{x}\to 0. \end{aligned}$$

For the model of a specific intensity we assume that the interior and exterior intensities are independent samples from the distributions p_{in} and p_{out}, respectively. The probability of observing $I(\vec{x})$ given $\phi(\vec{x})$ is given by

$$p(I(\vec{x})|\phi(\vec{x})) = \begin{cases} p_{\text{in}}(I(\vec{x})), & \text{if } \phi(\vec{x}) \le 0 \\ p_{\text{out}}(I(\vec{x}))^\lambda, & \text{if } \phi(\vec{x}) > 0 \end{cases},$$

where $\lambda \in \mathbb{R}$ can be chosen to increase or decrease the probability of an intensity belonging to the exterior in order to avoid leakage in areas where $p_{\text{out}}(I(\vec{x})) \approx p_{\text{in}}(I(\vec{x}))$. The value of $p_{\text{out}}(I(\vec{x}))$ lies between zero and one, and thus choosing $\lambda < 1$ will increase its value. This gives

$$\begin{aligned} \alpha \int_{\vec{x}\in\Omega} -\ln p(I(\vec{x})|\phi(\vec{x}))d\vec{x} = \alpha\Bigg(&\lambda \int_{\vec{x}\in\Omega} -\ln p_{\text{out}}(I(\vec{x}))H(\phi(\vec{x}))d\vec{x} \\ &+ \int_{\vec{x}\in\Omega} -\ln p_{\text{in}}(I(\vec{x}))[1-H(\phi(\vec{x}))]d\vec{x}\Bigg). \end{aligned}$$

The energy functional is thus given by

$$\begin{aligned} E_{\text{EM}}(\phi) := \alpha\Bigg(\lambda \int_{\vec{x}\in\Omega} -\ln p_{\text{out}}(I(\vec{x}))H(\phi(\vec{x}))d\vec{x} \\ + \int_{\vec{x}\in\Omega} -\ln p_{\text{in}}(I(\vec{x}))[1-H(\phi(\vec{x}))]d\vec{x}\Bigg) \\ + (1-\alpha)\int_{\Omega} \delta(\phi(\vec{x}))|\nabla\phi(\vec{x})|d\vec{x}. \end{aligned} \tag{2.55}$$

Now we note that in the derivation of the Chan–Vese level set equation in Section 2.3.1 the terms $|I(x)-c_i|^2$ act as constants in the derivation and thus we can replace them with the corresponding negative log likelihoods. This gives us the level set equation for the proposed model:

$$\begin{aligned} \phi_t(\vec{x},t) &= \Bigg[\alpha\big(\lambda \ln p_{\text{out}}(I(\vec{x})) - \ln p_{\text{in}}(I(\vec{x}))\big) \\ &\quad + (1-\alpha)\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right)\Bigg]|\nabla\phi|, \quad \text{in } (0,\infty)\times\Omega, \\ \phi(\vec{x},0) &= \phi_0(\vec{x}), \quad \text{in } \Omega, \\ \frac{\partial\phi}{\partial\hat{n}} &= 0, \quad \text{on } \partial\Omega. \end{aligned} \tag{2.56}$$

We have not yet discussed how the distributions p_{in} and p_{out} are to be chosen. In the case of segmenting objects, such as tumors with necrotic regions, each type of tissue displays intensities clustered close to one another with noise, and so in this thesis we choose to model the intensities of the interior region as coming from one of K_{in} Gaussian distributions, where K_{in} is the number of distributions. Similarly each observed intensity from the exterior region comes from one of K_{out} distributions. As we saw in Section 2.2 this implies that the distributions p_{in} and p_{out} are of the form

$$\begin{aligned} p_{\text{in}}(x) &= \sum_{k=1}^{K_{\text{in}}} \pi_k^{\text{in}} \mathcal{N}(x|\mu_k^{\text{in}}, \sigma_k^{\text{in}}), \\ p_{\text{out}}(x) &= \sum_{k=1}^{K_{\text{out}}} \pi_k^{\text{out}} \mathcal{N}(x|\mu_k^{\text{out}}, \sigma_k^{\text{out}}). \end{aligned} \tag{2.57}$$

These parameters can be estimated using expectation maximization based on the samples gathered during initialization of the level set function. This can be done by asking the user to draw a contour around the object of interest, see Figure 2.3.

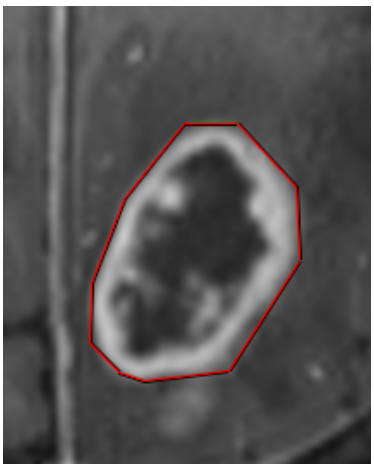


Figure 2.3: A contour is drawn around the region that should be segmented.

The contour $\mathcal{C}$ drawn by the user lies in a plane with normal $\vec{N}_\pi$. The level set function is initialized by first choosing a max distance $D_{\max}$ and a max sample distance D_{sample}. The idea is to initialize the level set function as an approximately signed distance function from the surface that is described by $\mathcal{C}$ in the plane Π, and by

$$\mathcal{C}_{\tilde{\Pi}} = \{\vec{x} \in \tilde{\Pi}; d(\text{proj}(\vec{x}, \Pi), \mathcal{C}) = d(\tilde{\Pi}, \Pi)\},$$

in parallel planes $\tilde{\Pi}$ with distance less than $D_{\max}$ from Π. In the two planes with distance exactly $D_{\max}$, ϕ should have value zero at all points inside the corresponding shrunk contour. The zero level set will describe a surface that is the user drawn contour in the plane it was drawn, and then shrinks inwards along the parallel planes until in a certain plane the volume is cut off, i.e. the entire interior of the contour is the surface. In order to avoid sampling incorrect points for the interior and exterior only points with a distance less than D_{sample} are sampled. The assumption is that the user draws the contour in the plane where the object is the largest, thus it can be assumed that in planes close to Π, the contour for the object is roughly equal to the contour drawn in Π, but slightly smaller than $\mathcal{C}$. Since the object to be segmented can have various shapes, this only holds true close to Π and so the max distance $D_{\max}$ should not be chosen large. The algorithm for initializing and sampling ϕ for the EM algorithm can be described as follows.

1. Compute the distance d_π between $\vec{x}$ and Π.
2. Compute the signed distance $d_\mathcal{C}$ between $\text{proj}(\vec{x}, \Pi)$ and $\mathcal{C}$, where $d_\mathcal{C} \leq 0$ for interior points and $d_\mathcal{C} > 0$ for exterior points. Then do the following:

   If $d_\mathcal{C} \leq 0$: Let $\phi(\vec{x}) = \max(d_\mathcal{C} + d_\pi, d_\pi - D_{\max})$.
   Else: Let $\phi(\vec{x}) = d_\mathcal{C} + d_\pi$.
3. If $d_\pi \leq D_{\text{sample}}$:

   If $\phi(\vec{x}) \leq 0$: Add $I(\vec{x})$ to the interior sample list.

Else: Add $I(\vec{x})$ to the exterior sample list.

We end by noting that while the distributions for the proposed method are decided at run time using the EM algorithm, a key problem lies in determining the parameters $\alpha, \lambda, K_{\text{in}}, K_{\text{out}}, D_{\max}$, and D_{sample}.

2.3.4 The Dice Coefficient

In order to test similarity between image segmentations, for example an automatic segmentation against a manual segmentation, the Dice coefficient is often used, [8]. Suppose we have manually segmented a volume and denote the set of points in the volume by M, similarly let A denote the set of points in the volume resulting from automatic segmentation. The Dice coefficient is then given by

$$D(A, M) = \frac{2|A \cap M|}{|A| + |M|} \in [0, 1], \tag{2.58}$$

where $|X|$ denotes the number of elements in the set X. If the two segmentations have no points in common the Dice coefficient equals 0, while all points in common imply a Dice coefficient equal to 1. Thus a high Dice coefficient implies greater similarity between the segmentations.

2.4 Coherent Propagation

Associated with the force $F(\vec{x}, t)$ in the level set equation (2.12) are internal and external forces. The external forces usually drive propagation by some force function that depends on the image intensity at a location. Internal forces depend on ϕ and are commonly incorporated as a function of the curvature of ϕ, and modifies the force to keep ϕ smooth. The issue that arises here is that certain regions might have a strong external force driving, for example, the contour to expand, which in turn cause the curvature to increase if neighboring regions have a lower external force. This causes the surface in high external force regions to contract in order to smooth the surface, followed by expansion as neighboring regions with a slower external force "catch up".

This process results in a "wiggling" behavior as certain points move faster than their neighbors, resulting in the curvature force moving them backwards even if eventually the entire neighborhood moves forwards. In [23] Wang et al. proposed an algorithm that avoids this phenomena by dividing the evolution of the interface into several periods, where each period begins by modifying the speed function and subsequently evolving the surface until convergence. At the start of each period a trend map is established for each grid point. During each period the force at a specific grid point must have the same sign as its trend, else the point is put to "sleep" until the sign of the force term changes, for example when neighboring points "catch up". The modified level set equation can be expressed as

$$\phi_t(\vec{x}, t) = w(F(\vec{x}, t))|\nabla\phi(\vec{x}, t)|,$$

where

$$w(F(\vec{x}, t)) = \begin{cases} F(\vec{x}, t), & \text{if } F(\vec{x}, t) \cdot \text{trend}(\vec{x}, t) > 0, \\ 0 & \text{if } F(\vec{x}, t) \cdot \text{trend}(\vec{x}, t) \leq 0. \end{cases}$$

The period ends naturally when $w(F(\vec{x},t)) = 0$ for all $\vec{x}$. This method is what is referred to as *coherent propagation*. In [23] it was noted that if only the CFL condition (2.24) was used to control step length, the surface could overshoot the true boundary when using coherent propagation. To solve this, the authors introduced a step length factor $s_{\max} < 1$ and a damping factor d. During coherent propagation, the step length is determined using the CFL condition and then multiplied by $s_{\max}$. After each period $s_{\max}$ is set to $d \cdot s_{\max}$. It was found that when $s_{\max} < 0.1$ the surface propagation can be stopped without affecting accuracy. It should be noted that coherent propagation is a heuristically chosen algorithm and its mathematical properties have not been studied. However, in [23] this method was found to produce results that were very similar to the unmodified level set method, while speeding up convergence by 10-100 times, depending on the type of segmentation. This results in implementations that perform segmentation in seconds rather than minutes. We will consider the case when the trend map is chosen to be equal for all grid points. This will cause the surface to alternate between expansion and contraction in each period. In the first period we have $\text{trend}(\vec{x},t) := -1$, only allowing negative values for F, i.e. expanding the contour. Then the trend alternates between 1 and -1. The algorithm consists of a main loop for the periods, and a sub loop for coherent propagation. The main loop is as follows.

1. Initialize the level set function ϕ and decide a maximum step length $s_{\max}$, a minimum step length $s_{\min}$, and a damping factor d. Set the period $p = 1$.
2. While $s_{\max} > s_{\min}$ do the following:
   - Let $\text{trend}(\vec{x}) := (-1)^p$ for all $\vec{x} \in \Omega$.
   - Perform coherent propagation.
   - Set $s_{\max} := s_{\max} \cdot d$.

The algorithm for coherent propagation is then:

1. Set $\text{status}(\vec{x}) = \text{active}\ \forall \vec{x} \in \Omega$.
2. Set iteration $= 0$ and choose a max number of iterations allowed, $n_{\max}$.
3. Clear changed_list.
4. For all $\vec{x} \in \Omega$ such that $\text{status}(\vec{x}) = \text{active}$, calculate $F(\vec{x})$.
5. Set $\Delta t = s_{\max} \cdot \frac{1}{max(F(\vec{x}))}$.
6. For all $\vec{x} \in \Omega$ such that $\text{status}(\vec{x}) = \text{active}$, do the following:

   If $\text{trend}(\vec{x}) = \text{sign}(F(\vec{x}))$:
   $\phi(\vec{x}) = \phi(\vec{x}) + F(\vec{x})|\nabla\phi(\vec{x},t)| \cdot \Delta t$.
   Add $\vec{x}$ to changed_list.
   Else:
   Set $\text{status}(\vec{x}) = \text{sleep}$.
7. For all $\vec{x}$ in changed_list set status of neighboring gridpoints to active.
8. Set iteration $=$ iteration $+ 1$.
9. If changed_list is not empty and iteration $< n_{\max}$ then go to step 3.

2.5 The Sparse Field Method

Introduced in [12] by Whitaker, the sparse field method (SPM) reduces the computational complexity of updating the level set function from $\mathcal{O}(N^3)$ to $\mathcal{O}(N^2)$, while maintaining the same level of accuracy. The SPM algorithm approximates a signed distance function by replacing the Euclidean distance with the Manhattan norm,

$$\|\vec{x}\|_1 := \sum_{i=1}^{n} |x_i|.$$

Suppose we have defined our Cartesian grid to have unit distance between grid points, so $\Delta x = \Delta y = \Delta z = 1$. This discrete grid defines a set of cells for which the zero level set interface S of the level set function ϕ passes. The grid points adjacent to S are called *active points*, and they form the *active set*. To compute finite differences, as described in Section 2.1.4, the neighboring grid points to the active set are used. Therefore at any point in time only the active points and their neighbors are relevant to the evolution of the surface.

Since ϕ is a distance function, the value of ϕ for the active set must lie within a range of values, called the *active range*. When the value at an active point moves out of this range it is no longer considered adjacent the surface, and thus is removed from the active set, and the grid points which have values moving into the active range are added to the active set. When the distance between gridpoints is one as chosen above, then the active grid point $\vec{x}$ should be removed from the active set when the value of $\phi(\vec{x})$ no longer is between $[-1/2, 1/2]$. When this occurs, a neighbor that has distance one from $\vec{x}$ will enter the active range, and thus be added to the active set.

It is shown in the appendix of [12] that the active set will always form a boundary between the interior and exterior of the volume, which implies that there will be no points in the active set where $|\nabla\phi| = 0$. This means that the finite difference approximation for the curvature term (2.25) will be defined for all active points.

Since grid points neighboring the active set have distance one from the neighboring active point, the active set can be used to update changes to the non-active adjacent grid points. These neighbors are defined as *layers*, $L_{\pm i}, i = 1, \ldots, N$, where i denotes the signed Manhattan distance to the closest active point. The positive values are used for the exterior region $\phi > 0$ (Ω^+), and the negative for the interior region $\phi \leq 0$ (Ω^-). The active set is denoted L_0. The N used should coincide with the neighborhood of grid points needed to compute the finite differences. When using the finite difference schemes discussed in Section 2.1.4, it is enough to set $N = 1$. Letting $\mathcal{G}$ denote the set of grid points we can express the layers as

$$L_i = \begin{cases} \{\vec{x} \in \mathcal{G}; i - \frac{1}{2} \leq \phi(\vec{x}) < i + \frac{1}{2}\} & \text{if } i < 0, \\ \{\vec{x} \in \mathcal{G}; \frac{1}{2} \leq \phi(\vec{x}) \leq \frac{1}{2}\} & \text{if } i = 0, \\ \{\vec{x} \in \mathcal{G}; i - \frac{1}{2} < \phi(\vec{x}) \leq i + \frac{1}{2}\} & \text{if } i > 0. \end{cases} \tag{2.59}$$

The algorithm is then as follows

1. Determine the layers using (2.59).
2. Update the level set function $\phi(\vec{x}, t)$ for all active grid points $\vec{x} \in L_0$.

3. Visit all grid points in the layers $L_i, i = \pm 1, \ldots, \pm N$ and update ϕ at the grid points based on the values of the inner layer $L_{i \mp 1}$ (by adding or subtracting one). If there is more than one neighbor in $L_{i \mp 1}$, use the value of the closest neighbor, i.e. the neighbor with minimum value for the interior, and the neighbor with maximum value for the exterior.

Chapter 3

Method

In this chapter the methods used to produce the results of the thesis are described. We begin with an overview of the development process for the proposed method. This is followed by a description of the test setup, how parameters were chosen, and finally the evaluation process for comparing the proposed method to the current method is provided.

3.1 Development of the Proposed Method

In the beginning of the thesis project Sectra described what goals they had for an improved segmentation tool. The discussion was centered around segmentation of objects with varying intensities. In these cases the current method they have performs inaccurate segmentations. However, it is in general very fast and simple to use. This is highly valued as the segmentation tool is used in clinical practice where high performance and user friendliness of Sectra's products is expected. The general focus for the thesis project was therefore to develop a segmentation tool that produced fast and accurate segmentations of inhomogeneous objects compared to the current method in a way that kept the user interaction simple and intuitive.

In order to better understand the current method and why it fails the theory for level set methods was studied. In Section 2.3 the reason for the inaccurate segmentation was identified and a new probabilistic force function was designed with inhomogeneity in mind. When studying how propagation of level set methods can be made faster, two algorithms were identified for providing order of magnitude speed increases; coherent propagation and the sparse field method. The existing implementation of the level set method uses the sparse field method combined with coherent propagation along with other optimizations in order to produce fast segmentations.

By developing a new segmentation method based on the level set method the underlying efficient algorithm for propagation of the surface could be used by replacing the external force function and modifying the user interaction and initialization. The proposed method presented in Subsection 2.3.3 was implemented in Sectra's Picture Archiving and Communication System (PACS). The software handles viewing medical images and has a vast framework of tools such as mouse and keyboard handling, drawing, and data structures. The code for

initialization from a user drawn contour was implemented in C# while the proposed level set method was written in C++ alongside the current implementation. It uses the numerical schemes described in Section 2.1.4 and the proposed method was implemented following the description in Subsection 2.3.3. The automated testing procedure was implemented in Matlab.

3.2 Testing Setup

In order to gather quantitative data on the performance of the proposed and existing method, a set of test cases were constructed. *The Cancer Imaging Archive* [25] is an open database containing anonymized medical images from thousands of exams. For this thesis we used *The Cancer Genome Atlas Glioblastoma Multiforme* (TCGA-GBM) data collection [26], which contains MRI images of patients with Glioblastoma, an aggressive form of brain cancer. The reason for this choice was that the tumors are often very inhomogeneous and irregular.

While the TCGA-GBM data collection contains hundreds of cases, 25 were chosen at random to conduct testing on since each case needed to be manually segmented, which was very time consuming. The manual segmentation was carried out by the author and done by viewing each case and drawing a contour around the border of the tumor in each image, using Sectra's software. An individual at Sectra with clinical experience then reviewed the images and after some adjustments deemed them acceptable manual segmentations. The contours were subsequently used to determine the set of voxels that belonged to the interior of the tumor.

Depending on what was to be tested, different metrics were gathered. However, the general common testing procedure can be described as follows:

1. Specify the cases to be tested.
2. For each case the three contours with largest area are identified and for each contour the following actions carried out:
   - Determine the longest line that can be drawn between two points on the contour's border which does not cross the contour.
   - Start timer and call level set method for existing method with line as argument.
   - Stop timer when propagation finishes. Save segmentation and time taken for existing method.
   - Start timer and call level set method for the proposed method with the contour as argument.
   - Stop timer when propagation finishes. Save segmentation and time taken for proposed method.
3. Analyze and save to file.

The reason for choosing the contours with largest area is to initialize the segmentation method in an area representative for the tumor. The three contours with largest area were analyzed in order to capture how small variations in initialization affected the segmentation.

All tests were carried out on a computer with an Intel Xeon E3-1240 v3 CPU, 16 gb of RAM, and an ATI FirePro V (FireGL V) GPU.

3.3 Parameter Tuning Process

In Subsection 2.3.3 a set of tunable parameters for the proposed method was identified. These are described in Table 3.1.

Parameter	Description
$K_{\text{in}} \in \mathbb{N}$	The number of components for the interior mixture model.
$K_{\text{out}} \in \mathbb{N}$	The number of components for the exterior mixture model.
$\alpha \in [0, 1]$	The weight of the external force in the speed function.
$(1 - \alpha) \in [0, 1]$	The weight of the internal (smoothing) force in the speed function.
$\lambda \in \mathbb{R}$	The weight of the exterior probability.
D_{max}	The max plane distance for initialized contour.
D_{sample}	The max plane distance for samples.

Table 3.1: Parameters for the proposed method.

It was also observed during implementation that allowing λ to have different values during the periods of coherent propagation greatly affected segmentation results. We recall from Section 2.4 that the propagation is divided into two types of periods. In one the contour described by the zero level set is only allowed to expand or stand still while in the second period only contraction or no movement is allowed. Setting $\lambda < 1$ during the expansion period (when $\text{trend}(\vec{x}) = -1$) implies that a higher internal probability for points is required to give a negative speed, i.e., cause expansion. However, using the same $\lambda < 1$ during the contraction period (when $\text{trend}(\vec{x}) = 1$) can cause the contour to contract too much as points are assigned positive external forces even if they have higher probability of belonging to the interior. Therefore using $\lambda = \lambda_{\text{expand}} < 1$ during expansion and $\lambda = \lambda_{\text{contract}} > 1$ during contraction seemed more promising than letting λ remain fixed. This translates to requiring high internal probability to expand and high exterior probability to contract.

Since the segmentation tool should be intuitive the user is not allowed to modify the parameters. Furthermore, no relationship between data and parameters was found during the development process. Therefore an automated testing procedure was developed to determine values for the parameters that perform well in general. Of the 25 cases manually segmented, 15 were chosen at random for determining the parameters. These are specified in Appendix A.

In the general testing procedure a set of automatic and manual segmentations are constructed. Let

$$P_{ij}, \quad \begin{cases} i = 1, 2, \ldots, 15 \\ j = 1, 2, 3 \end{cases}$$

denote the set of voxels belonging to the interior for the proposed method in case i for contour with the j^{th} largest area. The set of voxels belonging to the interior for the manual segmentation is given by M_i. Given these segmentations

how should the performance of the automatic segmentation be measured? The ultimate goal of the tool is to segment the object accurately in order to provide a trustworthy measure of volume. Therefore a good segmentation is one which is similar to the manual in location and volume. In Subsection 2.3.4 the dice coefficient was introduced as a metric of similarity. This however does not capture the differences in volume. This can be measured by computing

$$V(P_{ij}, M_i) = \frac{|P_{ij}|}{|M_i|}, \tag{3.1}$$

where $V(P_{ij}, M_i) = 1$ if the volumes are equal. It was determined that a combination of volume and segmentation similarity be used as a metric for the performance of the parameters. If only the volume is compared the segmentations can differ in location greatly which makes the accuracy of the volume measurement difficult to estimate in the absence of a manual segmentation. For parameter tuning the general automated testing procedure therefore saved the following quantity for case i given a set of parameters $\vec{P}$

$$W(i; \vec{P}) := \max_j \frac{1}{2} \left[\left(1 - D(P_{ij}, M_i)\right) + |1 - V(P_{ij}, M_i)| \right]. \tag{3.2}$$

The highest value recorded from the three contours was saved in order to produce parameters that performed well for all contours. The mean performance denoted $\mu(\vec{P})$ and standard deviation $\sigma(\vec{P})$ for a given set of parameters was then computed. At first it might seem natural to simply find parameters that minimize $\mu(\vec{P})$, however this can cause an optimum to have large variation and thus produce inconsistent segmentations with widely varying accuracy. Therefore, given an existing set of parameters $\vec{P}_1$, the following method was used to determine if a new set of parameters $\vec{P}_2$ should be considered better.

1. If $\mu(\vec{P}_2) < \mu(\vec{P}_1)$ and $\sigma(\vec{P}_2) < \sigma(\vec{P}_1)$, then $\vec{P}_2$ is better than $\vec{P}_1$.
2. Else if $\mu(\vec{P}_2) < \mu(\vec{P}_1)$ and $\sigma(\vec{P}_2) < \sigma(\vec{P}_1) + (\mu(\vec{P}_1) - \mu(\vec{P}_2))$, then $\vec{P}_2$ is better than $\vec{P}_1$.
3. Else if $\sigma(\vec{P}_2) < \sigma(\vec{P}_1)$ and $\mu(\vec{P}_2) < \mu(\vec{P}_1) + (\sigma(\vec{P}_1) - \sigma(\vec{P}_2))$, then $\vec{P}_2$ is better than $\vec{P}_1$.

Thus if $\vec{P}_2$ is *not* better than $\vec{P}_1$ then the μ cannot be reduced without increasing σ more than the amount that μ was reduced, or σ cannot be reduced without increasing μ by more than the amount that σ was reduced. This characterizes an optimal set of parameters as having consistent performance for all cases.

During the implementation phase a set of parameters for the proposed method were found that seemed promising. These were chosen to initialize the parameter tuning process. Running the automated testing procedure for a given set of parameters took approximately 5 minutes and given the amount of parameters it was determined not within the time budget to take the combinatoric approach of finding optimal parameters. The final method considers step sizes for the parameters and finds an optimum in the sense that varying any single parameter by at most 5 step lengths does not give a better set of parameters. Each parameter is modified in turn by increasing and decreasing its value by the minimum number of step lengths required to give an untested set of

parameters, but at most 5 step lengths. Then the automated testing procedure is run to determine if the modification is better. Once every parameter has been modified by 5 step lengths in each direction without finding an improved set of parameters the search ends. The parameters used for the tuning process are specified in Table 3.2.

Parameter	Start value	Step size
K_{in}	5	1
K_{out}	5	1
α	0.5	0.05
λ_{expand}	0.95	0.01
$\lambda_{\text{contract}}$	1.05	0.01
D_{max}	5	1
D_{sample}	1	0.5

Table 3.2: Parameters used for tuning process.

3.4 Evaluation

Once the parameter tuning process found optimal parameters the remaining 10 cases were used to compare the proposed method against the existing threshold based one, see Subsection 2.3.2. The cases used are specified in Appendix A. In the automated testing procedure the segmentation time for each contour was recorded along with metrics based on the segmentation results. The dice coefficient (2.58) and volume ratio (3.1) were computed in order to compare similarity. In order to have a meaningful interpretation of the mean volume ratio we introduce and instead study the volume error between an automatic segmentation X and manual segmentation M,

$$V_{\text{E}}(X, M) := |1 - V(X, M)| = \left|1 - \frac{|X|}{|M|}\right|. \tag{3.3}$$

Another interesting quantity is the amount of the automatic segmentation that was located inside the manual segmentation, which we henceforth refer to as the internal volume ratio,

$$V_{\text{I}}(X, M) = \frac{|X \cap M|}{|X|}, \tag{3.4}$$

where X is the segmentation for the proposed or existing method while M is the manual segmentation. Currently, Sectra's volume tool cannot modify the segmentation after it is completed apart from doing another segmentation that is added to the current one. This means that if the automatic segmentation leaks significantly the segmentation must be deleted and redone. Therefore a higher value of $V_I(X, M)$ is preferred, as this means it is possible to add to the current segmentation without needing to completely restart.

Chapter 4

Results

This chapter presents the main results of the thesis. First the results from the tuning process described in Section 3.3 are presented. Then each metric chosen for evaluation in Section 3.4 is presented. As described in Section 3.2 each case was segmented using the three largest contours. These are labeled as contour 1,2, and 3, ordered according to area size. The two methods are compared based on the average metric achieved for each case and the mean deviation within each case. We end with a series of figures comparing segmentation for three cases chosen based on metric abnormalities.

Remark 4.1. When presenting the performance for different cases the x-axis is labeled as "Case ID"; this refers to the ID assigned to each case in Appendix A, Table A.2.

4.1 Parameter Tuning

The tuning process found a set of parameters satisfying the optimality condition described in Section 3.3 after 70 iterations of parameter modification and testing. A total of 155 iterations took place in order to check up to 5 step lengths in each directions. This took approximately 13 hours. In Table 4.1 the optimal parameters are compared to the start values. The best parameters found were subsequently used when comparing the proposed method to the existing one.

Parameter	Start value	Optimal value	Net change
K_{in}	5	5	0
K_{out}	5	7	+2
α	0.5	0.5	0
λ_{expand}	0.95	0.95	0
$\lambda_{\text{contract}}$	1.05	1.09	+0.04
D_{max}	5	6	+1
D_{sample}	1	1	0

Table 4.1: Start parameters and final parameters found in tuning process.

Each iteration consisted of modifying a parameter and then testing the performance. In Figure 4.1 the performance metric and standard deviation is plotted for iterations where new parameters were chosen. Recall from Section 3.3 that better parameters lowered the mean performance metric or standard deviation.

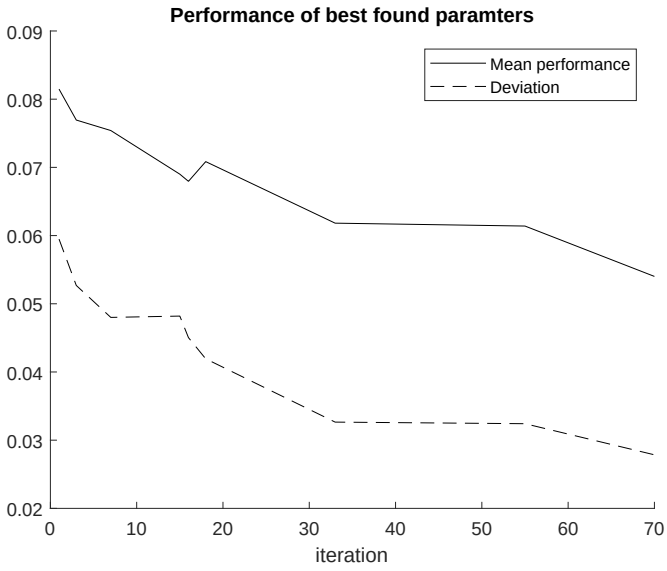


Figure 4.1: Performance and deviation at iterations where better parameters were found.

4.2 Segmentation Time

We begin by looking at the performance across all segmentations. In Table 4.2 the mean segmentation time and standard deviation of all segmentations are compared.

Method	Mean	Deviation
Existing	0.70	0.24
Proposed	0.77	0.17

Table 4.2: Segmentation times in seconds for all contours.

In Figure 4.2 each case is compared based on the mean segmentation time for the three segmentation performed in each case.

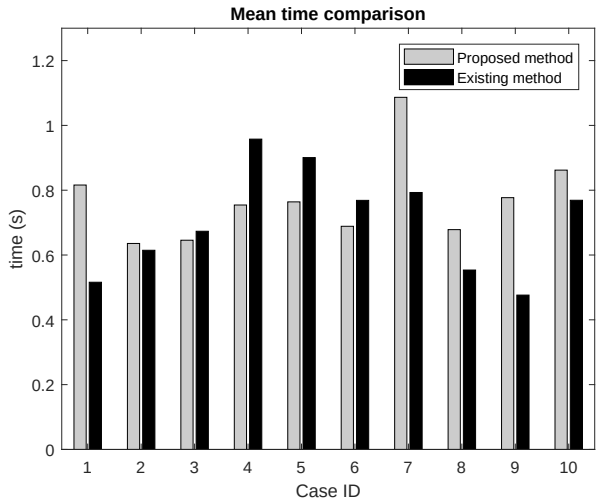


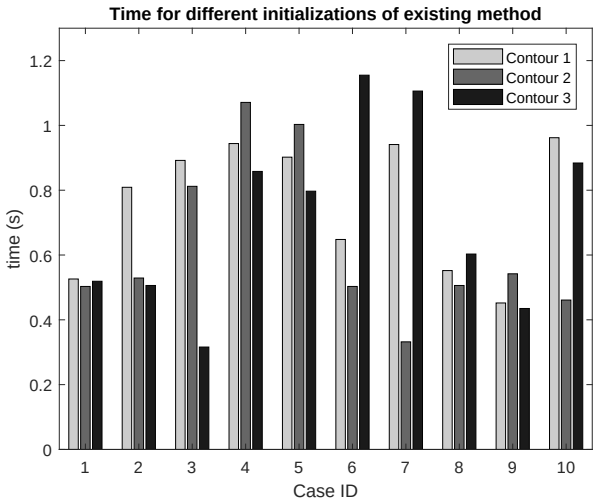
Figure 4.2: Mean segmentation times for proposed and existing method.

Next we look at how segmentation varied in each case. In Table 4.3 the mean deviation of segmentation time in each case is shown.

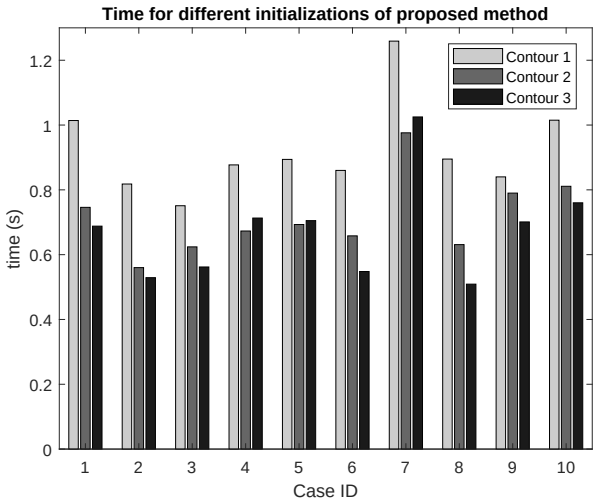
Method	Mean deviation
Existing	0.18
Proposed	0.14

Table 4.3: Mean segmentation time deviation for each case.

In the following figures the segmentation time for each contour is presented. Figure 4.3a shows the times for the existing method while Figure 4.3b shows the times for the proposed method.



(a) Segmentation times using existing method.



(b) Segmentation times using proposed method.

Figure 4.3: Segmentation times for each contour.

4.3 Dice Coefficient

Next we look at the similarity between automatic and manual segmentations. This is done by computing the dice coefficient (2.58). In Table 4.4 the mean dice coefficient and standard deviation of all segmentations are compared.

Method	Mean	Deviation
Existing	0.57	0.30
Proposed	0.84	0.05

Table 4.4: Dice coefficient for all contours.

In Figure 4.4 each case is compared based on the mean dice coefficient for the three segmentations performed in each case.

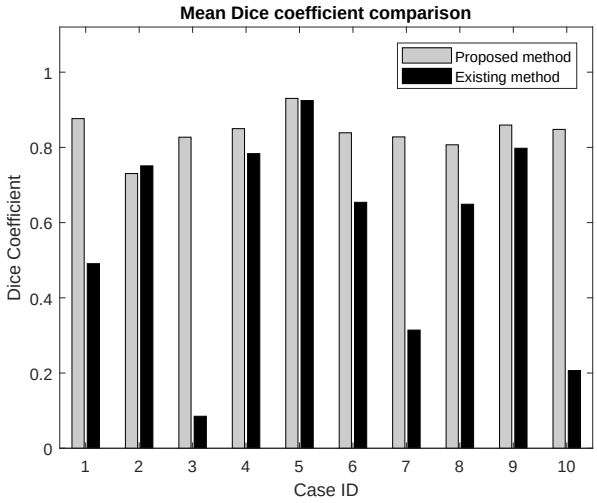


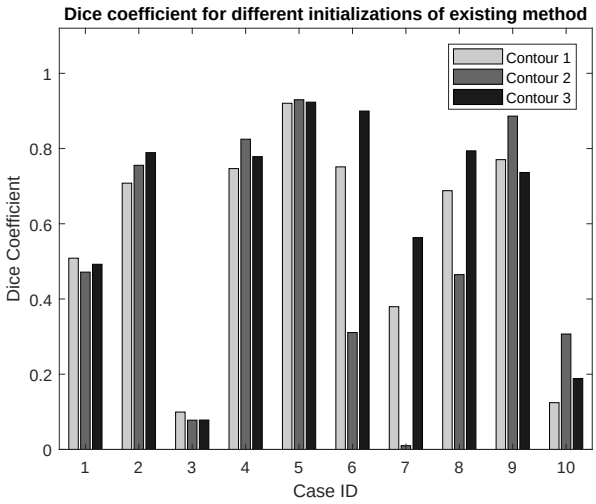
Figure 4.4: Mean dice coefficient for proposed and existing method.

As for the comparison of segmentation time we now look at how the dice coefficient varied in each case. In Table 4.5 the mean deviation of the dice coefficient in each case is presented.

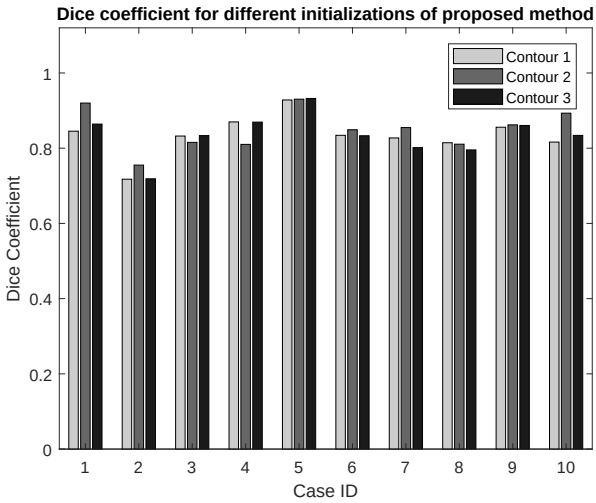
Method	Mean deviation
Existing	0.10
Proposed	0.02

Table 4.5: Mean dice coefficient deviation for each case.

We end by displaying the dice coefficient for each contour. In Figure 4.5a the dice coefficients for the existing method is shown while Figure 4.5b shows the dice coefficients for the proposed method.



(a) Dice coefficients using existing method.



(b) Dice coefficients using proposed method.

Figure 4.5: Dice coefficients for each contour.

4.4 Volume Error

In Section 3.4 the volume error (3.3) was introduced to measure the difference in volume between the automatic and manual segmentation. In Table 4.6 the mean volume error and standard deviation of all segmentations are compared.

Method	Mean	Deviation
Existing	1.11	2.19
Proposed	0.21	0.11

Table 4.6: Volume error for all contours.

Figure 4.6 compares the mean volume error for the three segmentations performed for each method. As the existing method had cases where the error was extremely large the bar plot is cut at 0.6, which corresponds to 60% volume difference, in order to be able to compare the differences in cases where the errors are smaller.

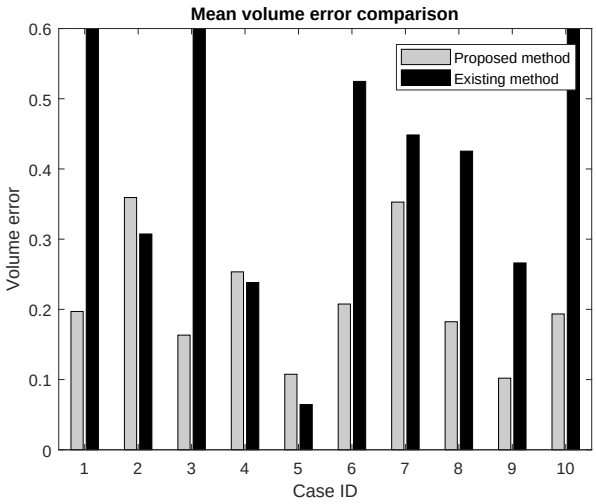


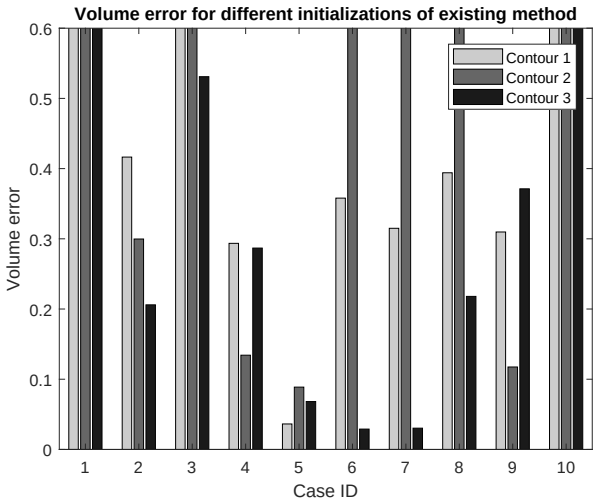
Figure 4.6: Mean volume error for proposed and existing method.

Next we look at the mean volume error deviation in each case. This is shown in Table 4.7.

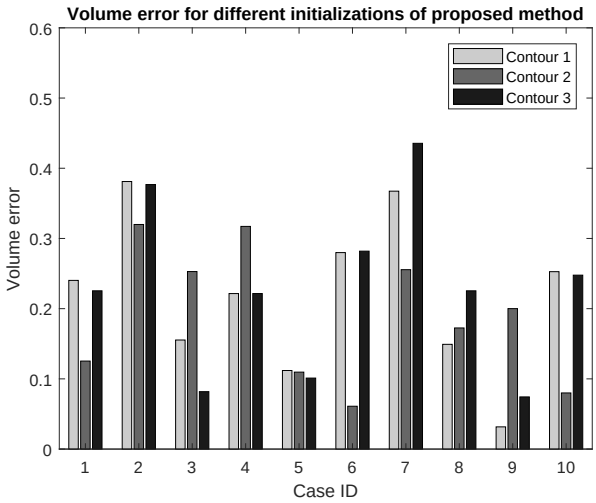
Method	Mean deviation
Existing	0.94
Proposed	0.07

Table 4.7: Mean volume error deviation for each case.

The volume error for each segmentation is presented in Figure 4.7a for the existing method and in Figure 4.7b for the proposed method. Again we cut the axis at 0.6 in order to compare the results.



(a) Volume errors using existing method.



(b) Volume errors using proposed method.

Figure 4.7: Volume errors for each contour.

To investigate if there is a connection between volume errors and segmentation times the mean segmentation time for segmentations having a maximal volume error are plotted in Figure 4.8.

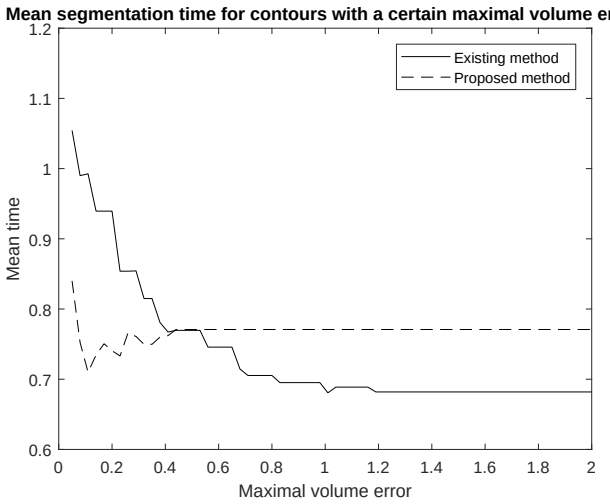


Figure 4.8: Mean segmentation times for segmentations with different maximal volume errors.

4.5 Internal Volume Ratio

In Section 3.4 the internal volume ratio (3.4) was introduced as a measure of the percentage of automatic segmentation's volume located inside the manual segmentation. In Table 4.8 the mean internal volume ratio and standard deviations of all segmentations are shown.

Method	Mean	Deviation
Existing	0.70	0.36
Proposed	0.93	0.08

Table 4.8: Internal volume ratio for all contours.

In Figure 4.9 the mean internal volume ratio for the three segmentations performed for each method is compared.

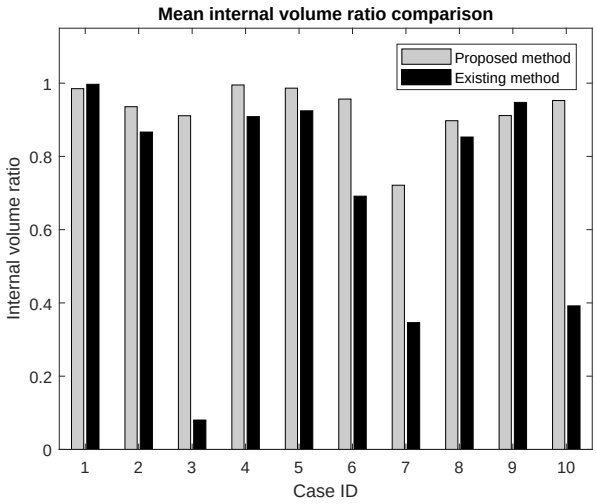


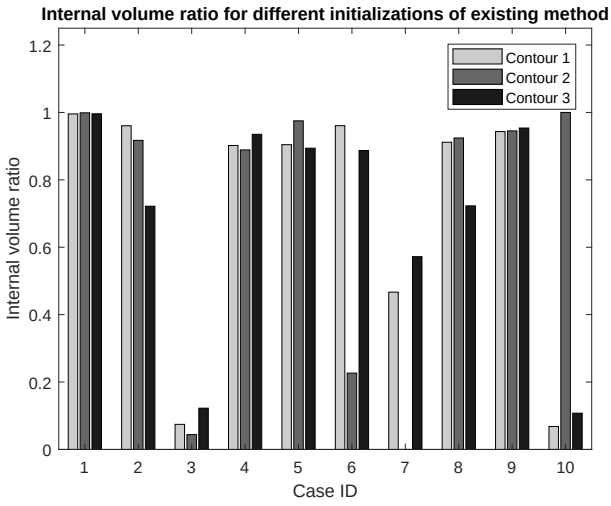
Figure 4.9: Mean internal volume ratio for proposed and existing method.

The mean deviation within each case is shown in Table 4.9.

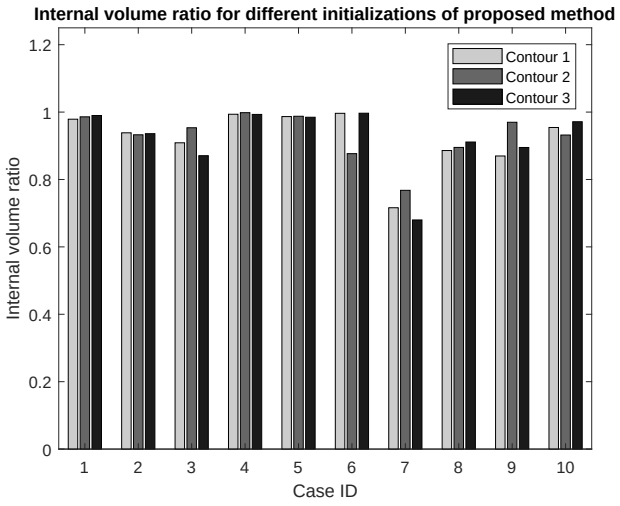
Method	Mean deviation
Existing	0.16
Proposed	0.03

Table 4.9: Mean internal volume ratio deviation for each case.

Finally, the internal volume ratio for each contour is presented in Figures 4.10a and 4.10b for the existing and proposed method, respectively.



(a) Internal volume ratios using existing method.



(b) Internal volume ratios using proposed method.

Figure 4.10: Internal volume errors for each contour.

4.6 Segmentation Comparison

We conclude by comparing segmentations for a few cases. As segmentation in three dimension is difficult to illustrate properly we will show the image where segmentation is initialized and then the resulting segmentation in a few locations. Looking at Figure 4.4 where the mean dice coefficients are compared we see that both methods achieve the highest dice coefficient in case 5. In Figure 4.11 segmentation with the existing method and proposed method is displayed. The red contour visible in each image is the manual segmentation while the blue contour is the automatic segmentation.

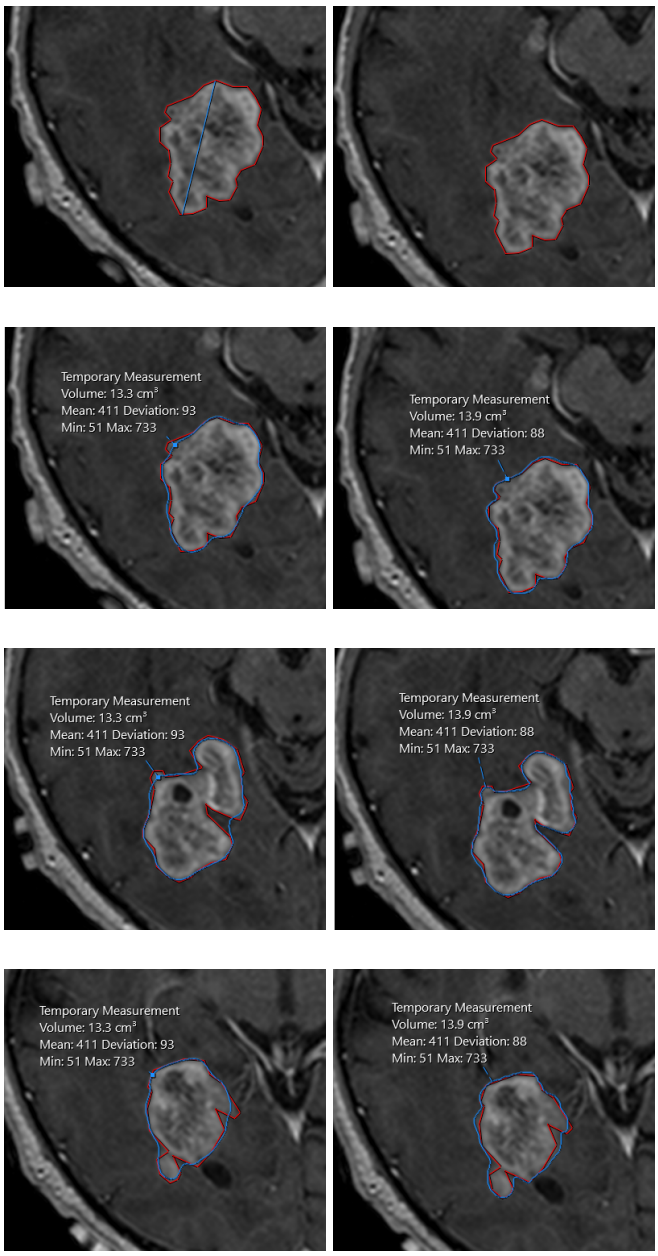


Figure 4.11: Comparison of segmentation for case 5. Left column shows existing method, right column show the proposed method. The red contour is the manual segmentation. The existing method is initialized from the blue line and the proposed method from the contour.

Next we look at Figure 4.6 where the mean volume error for the proposed method is largest in case 2. This is illustrated in Figure 4.12.

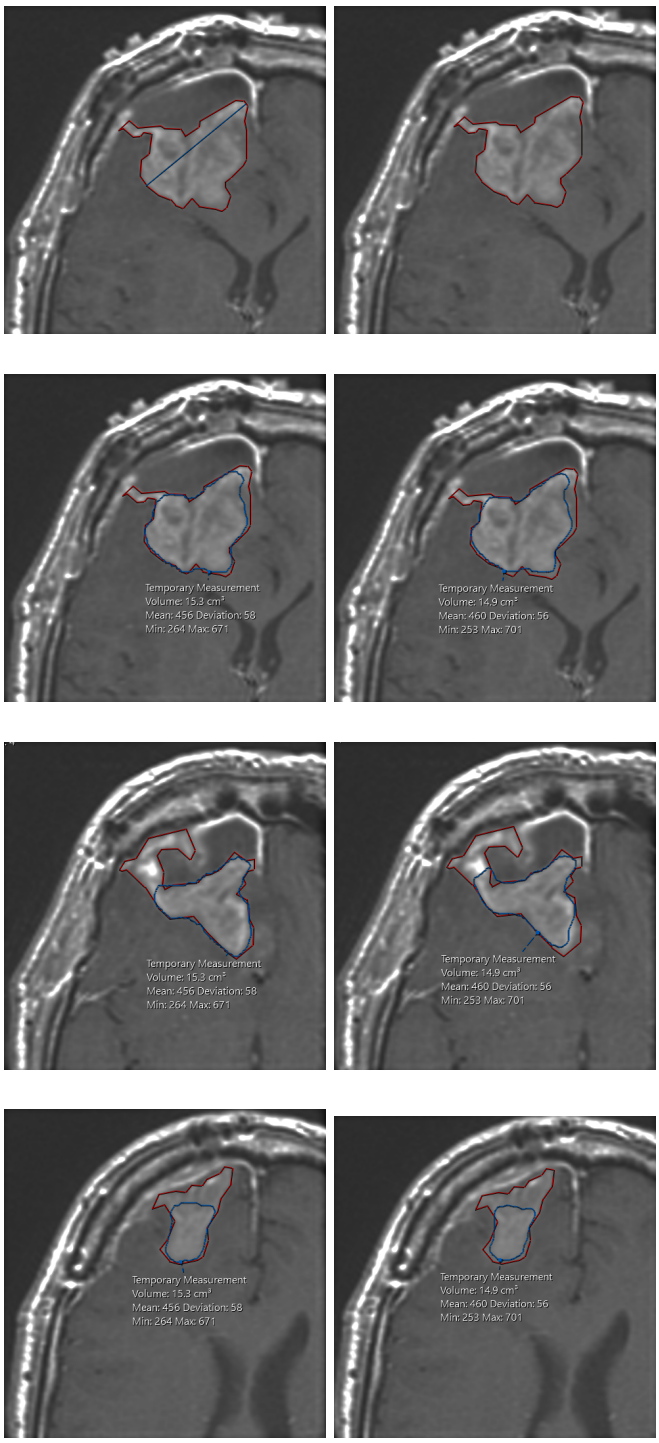


Figure 4.12: Comparison of segmentation for case 2. Left column shows existing method, right column show the proposed method. The red contour is the manual segmentation. The existing method is initialized from the blue line and the proposed method from the contour.

Finally, we note that the mean volume error is extremely large in Figure 4.6 for the existing method in case 10 while the proposed method has a much lower volume error. In Figure 4.13 the segmentations are again compared.

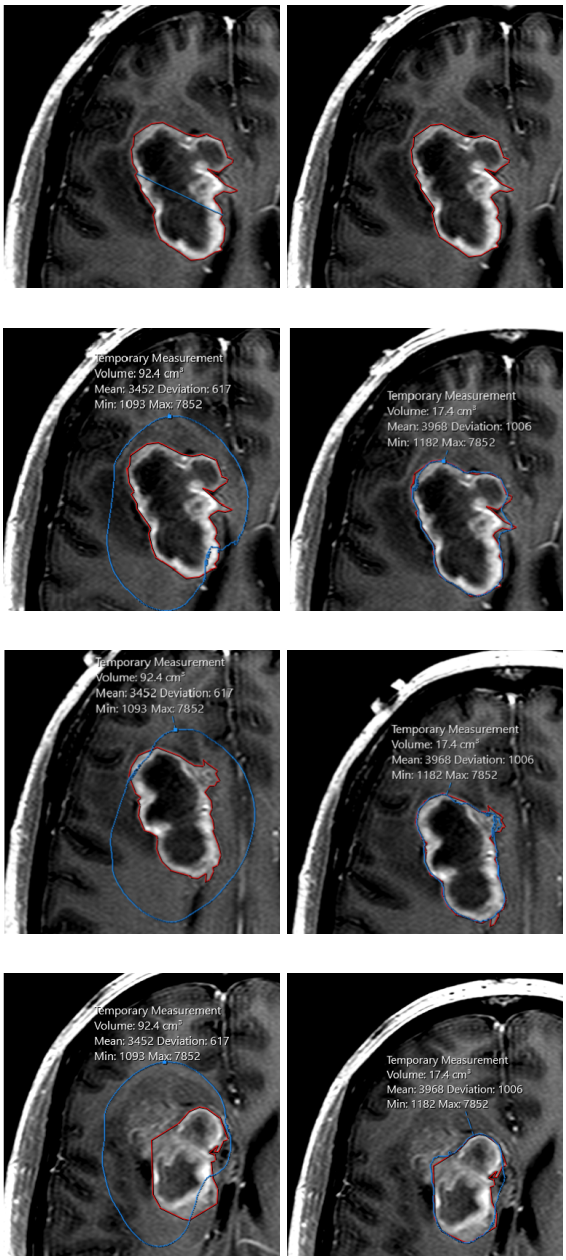


Figure 4.13: Comparison of segmentation for case 10. Left column shows existing method, right column show the proposed method. The red contour is the manual segmentation. The existing method is initialized from the blue line and the proposed method from the contour.

Chapter 5

Discussion

This chapter will discuss the results and the method. We will end with a discussion on future work and improvements that could be considered.

5.1 Results

In Table 4.1 we see that several starting parameters were not modified during the tuning process. This coupled with the fact that the performance for the starting parameters was already quite good, as seen in Figure 4.1, could indicate that the starting parameters happened to be close to the optimum, the cases were not varying enough, or that more cases should have been used. It is less likely that many local optimum exist as the method attempted to vary each parameter greatly at the end of the tuning process, although it is possible that a better optima could be attained by modifying more than one parameter simultaneously or by using a smaller step size. The results, however, clearly indicate, through the lens of the metrics considered, that the proposed method is better suited for segmentation of inhomogeneous tumors than the existing one. Among all the metrics considered the mean performance of the proposed method was improved while at the same time decreasing the deviation. Furthermore, the mean intra-case deviation was also lowered.

5.1.1 Segmentation Time

In Table 4.2 we observe that the mean segmentation time is increased by 0.07 seconds, or 10%, for the proposed method while reducing the deviation by 0.07 seconds. The proposed method performed more consistently within each case, with a mean intra-case deviation of 0.14 compared to 0.18 for the existing method. While the existing method does perform very fast segmentations in certain cases, as seen in figure 4.3a (e.g., case 1, case 3 contour 3, and case 7 contour 2) the large variation in segmentation times results in a mean time comparable to the proposed method. On the other hand, in Figure 4.3b all segmentations take at least 0.5 seconds for the proposed method. The absence of faster segmentations can be explained by the fact that the proposed method requires considerably more computation time during initialization, regardless of the subsequent propagation time.

Furthermore, Figure 4.8 shows that the fast segmentations occur for the existing method in cases with a large volume error while the proposed method performs consistently for different maximal volume errors. For example, the mean segmentation time for cases with maximal volume error 20% is 0.94 seconds for the existing method and 0.74 seconds for the proposed method. It is only when segmentations with volume errors exceeding 40% are considered that the existing method becomes faster. However, in such cases it is likely that the segmentation will need to be redone which will lead to a larger total segmentation time.

5.1.2 Similarity

In Section 4.3 and 4.4 the dice coefficient and volume error were compared to determine the level of similarity to the manual segmentation. In Table 4.4 the dice coefficient for the proposed method is 0.27 (48%) higher than for the existing one while having a deviation of 0.05 compared to 0.3 for the existing method. The latter can be attributed to case 1,3,7, and 10 as seen in Figure 4.4 where the dice coefficient for the existing method are extremely low. In Figure 4.5a we see that the existing method has a dice coefficient of almost zero for contour 2 in case 7. Looking at Figure 4.7a in Section 4.4 we see that the low dice coefficient can be explained by the segmentation leaking, as illustrated by the large volume error.

In Figure 4.5b we see that the proposed method has a much more consistent dice coefficient across the cases and within. The lowest performance is attained in case 2. A visual inspection in Figure 4.12 show that parts of the tumor are missed. A notable aspect of case 2 is that it consists of only 57 images and furthermore the tumor changes shape quickly throughout the images. Thus even if the tumor is relatively homogeneous both the existing and proposed method have difficulty performing accurate segmentations as the smoothing force becomes large due to the curvature becoming large as the segmentation attempts to propagate with great variation in neighboring planes.

We also observe in Figure 4.4 that both methods achieve their highest dice coefficient in case 5. In Figure 4.6 we note that the proposed method and existing method have very low volume errors in this case. In Figure 4.11 we see that both methods are close to the manual segmentation, the tumor is quite homogeneous and furthermore the case has 192 images causing the tumor not to quickly change between neighboring planes as the images are taken more densely.

Due to the cases where the current method fails to detect the boundary the resulting mean volume error is extremely large, as seen in Table 4.6. On the other hand, the proposed method has a more consistent error of 0.21 with deviation 0.11. In Figure 4.6 we see that the existing method has a very large volume error compared to the proposed method in several cases. This was visually inspected in Figure 4.13. Here we see that the tumor consists of a very dark and bright region while the surrounding tissue lies between these values. The existing threshold method fails completely as one might expect at establishing an appropriate threshold for the tumor and leaks severely across the boundary.

From these results it is no surprise that the internal volume ratio for the proposed method is considerably higher than for the existing one. In Table 4.8

we see that on average only 70% of the automatic segmentation is located within the manual segmentation for the existing method with a deviation of 36%. For the proposed method 93% is located within the manual segmentation with a deviation of only 8%.

5.2 Method

The tuning process was quite rudimentary and running it multiple times with different starting parameters and step sizes could have resulted in better parameters. While only 15 cases were used for the tuning process and 10 for the evaluation, performing multiple segmentations per case resulted in more test cases while also providing insight into how different initializations affected the metrics. It is possible that the manual segmentations affected the result and it should be noted that even among radiologists manual segmentations can differ greatly.

The approach of using the largest contours for initializations provided a convenient way of performing a large number of objective segmentations. It also means that increasing the number of cases to be tested only requires performing manual segmentations with no additional work required to incorporate them into the automated testing procedure. Furthermore, looking at several different metrics and how these varied within the cases gave insight into the qualitative performance of the methods through quantitative data.

Reproducibility of the results should be mentioned. While the proposed method is presented in its entirety the various optimizations Sectra's implementation of the level set method performs are not. This means that attempting to use the proposed method with coherent propagation and the sparse field method would not give the same segmentation times. Furthermore, the way in which the existing method determines the exact threshold to use is not disclosed making it difficult to perform the same comparison. However, the underlying flaw in the threshold method should give similar results regardless of how they are chosen, as the model cannot handle multiple interior intensity distributions spread apart.

5.3 Further Work and Improvements

One of the most interesting results was that the proposed method achieved the highest and lowest dice coefficient for tumors that were very homogeneous but where the lower result came from a case where there were few images with a large distance between them resulting in great variation in tumor shape between neighboring planes. It would therefore be very interesting to see methods designed to account for this in the smoothing curvature term. One approach could be to artificially add images by interpolation, another would be to consider curvature in each plane separately.

As Sectra's volume segmentation tool is used in many different types of exams a crucial step before replacing the existing method with the proposed one is to investigate how the method performs in different types of medical images of in a variety of exams.

An improvement that could be considered for any type of volume segmentation tool is the ability to modify the segmentation after it is done. Often the segmentation covers a large part of the object but some part has been missed. For single images there exists a variety of so called "nudge tools" that allow smooth deformation of the contour. However, it is considerably more difficult to allow the user to nudge the boundary of a volume in one plane that deforms the boundary surrounding the plane in a manner the user can easily control.

We also observe that the proposed method attempts to create a Gaussian mixture model for the interior and exterior with no temporal information. Thus if both the exterior and interior display a similar distribution the method can fail. Future work could examine how this can be handled, for example by considering how intensities correlate to the surrounding or smoothing the image in order to remove distributions arising from noisy images.

Another approach to improve segmentation results is to use a GPU-accelerated level set method. This could potentially improve segmentation times significantly and allow for multiple automatic segmentations to be performed and then compared in a structured manner in order to improve the final segmentation. For example different weights on smoothing could be used to attempt detection of leakage.

Finally, we note that different types of MRI images are usually taken in an exam with the same frame of reference, i.e. the body locations are identical in the images. Since these images highlight different organic structures, such as water or fat, it might be possible to build a statistical model that utilizes this. For example, the border might be clearer in a certain type of image and a speed function could therefore be used that always uses the probabilities from the MRI images that give the largest absolute speed. This corresponds to using the probability from the image type where the interior and exterior probability differ the greatest.

Chapter 6

Conclusions

We are now ready to answer the research questions posed at the beginning of the thesis:

> *How is the level set method used for image segmentation?*
> By embedding the segmentation boundary implicitly as the zero level set of a function, it is possible to construct various energy functionals that depend on the segmentation and are minimized when the zero level set is located at the true boundary of the object. Using the Euler–Lagrange equation a gradient descent method can be constructed that finds a minimizer to the functional.
>
> *Can the level set method be used to develop a segmentation algorithm that performs better on MRI images of heterogeneous brain tumors than the existing method?*
> By modifying the initialization of the level set function and the probabilistic model used in [20] a level set method can be constructed that segments heterogeneous brain tumors in MRI images more accurately than the existing method with regard to the dice coefficient, volume error, and internal volume ratio.
>
> *How does segmentation time for the new method compare to the existing method?*
> While the existing method can perform fast segmentations in certain cases, and is 10% faster than the proposed method overall, the absolute difference of 70 milliseconds is arguably negligible. Furthermore, when considering cases where the volume error is at most 40%, the proposed method performs faster segmentations than the existing method, at the cost of higher user interaction time, due to the placement of a contour instead of a line.

Let us end by remarking on a few more subtle aspects of the material presented in this thesis. First, we note that new segmentation methods can be constructed that can improve results greatly by simply modifying the level set speed function. Thus the underlying efficient implementation of for example the sparse field method and coherent propagation can be reused. As initialization is independent from propagation it is also possible to modify initialization and user interaction without affecting the propagation algorithm. Secondly, we note that even with improved level set methods for image segmentation such as the one presented

in this thesis, the segmentation will often miss parts of the tumors and it would be very beneficial for any segmentation algorithm to include a tool that allows deforming the final segmentation. Lastly, we note that the testing procedure used in this thesis could be incorporated into an automated testing framework used by developers to quickly check what effect changes to the segmentation algorithm have. This could greatly ease the development of different methods for segmentation as the effect for the end user can be determined with greater confidence.

Appendix A

Test Data

The MRI images used in this thesis can be retrieved from [26]. In all cases it is the T1-weighted MRI images were used. Below the specific case names for evaluation are listed alongside the ID they were given for testing. Each exam has a varying amount of image slices taken however each image has resolution 512×512.

Case name	Image count
TCGA-06-0182	79
TCGA-06-0238	84
TCGA-06-0176	60
TCGA-06-0189	76
TCGA-06-0210	31
TCGA-08-0244	124
TCGA-27-2519	192
TCGA-12-0620	128
TCGA-27-1833	192
TCGA-27-2518	192
TCGA-06-0216	30
TCGA-06-0213	56
TCGA-02-0037	124
TCGA-06-0185	38
TCGA-06-0190	71

Table A.1: Test cases for parameter tuning.

Case ID	Case name	Image count
1	TCGA-02-0034	124
2	TCGA-06-0214	57
3	TCGA-06-0221	60
4	TCGA-06-0238	84
5	TCGA-27-2523	192
6	TCGA-27-2526	80
7	TCGA-27-2527	192
8	TCGA-06-0187	41
9	TCGA-08-0529	124
10	TCGA-02-0027	124

Table A.2: Test cases for evaluation.

References

[1] R. Courant, K. Friedrichs, and H. Lewy, *On the partial difference equations of mathematical physics*, IBM Journal of Research and Development, 11(2), 215–234, 1967.

[2] J.B. MacQuuen, *Some Methods for classification and Analysis of Multivariate Observations*, Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, 1, 281–297, 1967.

[3] A.P. Dempster, N.M. Laird, and D.B. Rubin, *Maximum Likelihood from Incomplete Data via the EM Algorithm*, Journal of the Royal Statistical Society, 39(1), 1–38, 1977.

[4] J. Wu, *On the Convergence Properties of the EM Algorithm*, The Annual of Statistics, 11(1), 95–103, 1983.

[5] L. Hörmander, *The analysis of linear partial differential operators I*, Springer, 1983.

[6] S. Osher and J. Sethian, *Fronts Propagating with Curvature Dependent Speed: Algorithms Based on Hamilton–Jacobi Forumlations*, Journal of Computational Physics, 79, 12–49, 1988.

[7] C. Hirsch, *Numerical Computation of Internal and External Flows*, John Wiley & Sons, 1990.

[8] A.P. Zijdenbos, B.M. Dawan, R.A. Margolin, A.C. Palmer, *Morphometric Analysis of White Matter Lesions in MR Images: Method and Validation*, IEEE Transactions on Medical Imaging, 13(4), 1994.

[9] D. Adalstein and J.A. Sethian, *A Fast Level Set Method for Propagating Interfaces*, Journal of Computational Physics, 118(2), 269–277, 1995.

[10] H.K. Zhao, T. Chan, B. Merriman, and S. Osher, *A variational level set approach to multiphase motion*, Journal of Computational Physics, 127, 179–195, 1996.

[11] V. Caselles, R. Kimmel, G. Sapiro, *Geodesic Active Contours*, International Journal of Computer Vision, 22(1), 61–79, 1997.

[12] R.T. Whitaker, *A Level-Set Approach to 3D Reconstruction from Range Data*, International Journal of Computer Vision, 29(3), 203–231, 1998.

[13] J.A. Sethian, *Level Set Methods and Fast Marching Methods*, Cambridge University Press, New York, 1999.

[14] Y.L. Zhang, K.S Yeo, B.C. Khoo, and C. Wang, *3D Jet Impact of Toroidal Bubbles*, Journal of Computational Physics, 166, 336–360, 2001.

[15] T.F. Chan and L.A. Vese, *Active contours without edges*, Journal of Transactions on Image Processing, 2(10), 266–277, 2001.

[16] S. Osher and N. Paragios, *Geometric Level Set Methods*, Springer–Verlag, New York, 2003.

[17] A. Lefohn, J. Cates, R. Whitaker, *Interactive, gpu-based level sets for 3D segmentation*, Proceedings of the MICCAI conference, 564–572, 2003.

[18] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*, Springer–Verlag, New York, 2003.

[19] C.M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, 2006.

[20] D. Cremers, O. Fluck, M. Rousson, and S. Aharon, *A Probabilistic Level Set Formulation for Interactive Organ Segmentation*, Medical Imaging, 65120V, 2007.

[21] L.C Evans, *Partial Differential Equations*, American Mathematical Society, 2010.

[22] A. Mitiche and I.B. Ayed, *Variational and Level Set Methods in Image Segmentation*, Springer, 2010.

[23] C. Wang, H. Frimmel, and Ö. Smedby, *Fast level-set based image segmentation using coherent propagation*, American Association of Physicists in Medicine: Medical Physics, 41(7):073501, 2016.

[24] S. Borman, *The Expectation Maximization Algorithm, A short tutorial*, Available at: https://www.cs.utah.edu/~piyush/teaching/EM_algorithm.pdf, Accessed 18 Apr. 2018.

[25] *The Cancer Imaging Archive*, https://wiki.cancerimagingarchive.net/display/Public/Wiki.

[26] *The Cancer Genome Atlas Glioblastoma Multiforme data collection*, https://wiki.cancerimagingarchive.net/display/Public/TCGA-GBM.