Multi-resolution Image Segmentation using Geometric Active Contours

by

Po-Yan Tsang

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master in Applied Science
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2004

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Po-Yan Tsang

Abstract

Image segmentation is an important step in image processing, with many applications such as pattern recognition, object detection, and medical image analysis. It is a technique that separates objects of interests from the background in an image. Geometric active contour is a recent image segmentation method that overcomes previous problems with snakes. It is an attractive method for medical image segmentation as it is able to capture the object of interest in one continuous curve.

The theory and implementation details of geometric active contours are discussed in this work. The robustness of the algorithm is tested through a series of tests, involving both synthetic images and medical images. Curve leaking past boundaries is a common problem in cases of non-ideal edges. Noise is also problematic for the advancement of the curve. Smoothing and parameters selection are discussed as ways to help solve these problems.

This work also explores the incorporation of the multi-resolution method of Gaussian pyramids into the algorithm. Multi-resolution methods, used extensively in the areas of denoising and edge-selection, can help capture the spatial structure of an image. Results show that similar to the multi-resolution methods applied to parametric active contours, the multi-resolution can greatly increase the computation without sacrificing performance. In fact, results show that with successive smoothing and sub-sampling, performance often improves.

Although smoothing and parameter adjustment help improve the performance of geometric active contours, the edge-based approach is still localized and the improvement is limited. Region-based approaches are recommended for further work on active contours.

Acknowledgements

My sincere gratitude to my supervisor Ed Jernigan, for his guidance and for giving me the opportunity to explore image processing. Thank you to people in the lab: Rishi, Mark, and Fu Jin for the discussions and help. Thank you Christine for sharing the frustrations. Thank you Phil, for a bit of everything in the past two years. I dedicate this thesis to my parents, for their unwavering support and encouragement.

I would like to thank Fan Ding of the University of Wisconsin, Madison, for allowing me to expand his code on level sets. Also thanks to Raphael Ronen and Dr. Brian Rutt at the Robarts Research Institute for letting us acquire spinal MRI images; to Clinton Chau at Merge Efilm for brain MRI images.

Contents

List of Figures

Chapter 1

Introduction

Image segmentation, an image processing technique to separate objects from the background in an image, is an important step in image processing with many applications such as machine vision, face recognition, and medical image analysis.

In machine vision, robots can rely on cameras to give information about the surroundings, such as identifying proper route or possible obstacles. Highway toll systems use cameras to extract car license plates, and in manufacturing, images help discover possible defects in products such as cracks or breakages.

These days, as terrorist threats loom and security systems become more in demand, face recognition is an important area of research as computers help matches people to database profiles through identifying face features. To be able to isolate face features such as eyes, mouth and eye brows are difficult widely researched problems.

As health care systems and hospitals become more digitalized, medical image processing plays a more important role. As an intermediate step, image segmentation is crucial in image registration and classification problems. Furthermore, medical image segmentation offers help in diagnosis and treatment in many situations. First of all, through 2-dimensional segmentation, a 3-dimensional model can be reconstructed, which helps with visualization for diagnosis and

aid in surgery. Also, doctors can estimate dimension (volume, area, length) of the object of interest such as a tumor such that radiation therapy can be automated. With the trend of the digitization of medical data, segmentation also helps form a digital atlas of the patient's anatomy, thus eliminating the need for patient specific tools [12].

1.1 Medical Imaging

Medical imaging evolved from the discovery of x-rays to the newest magnetic resonance image (MRI). Commonly used techniques currently include x-ray, computer tomography (CT), ultrasound, MRI, and positron emmision tomography (PET). This section will briefly discuss the nature of some of these images and the technology behind them.

1.1.1 X-ray

X-ray is the first and oldest medical imaging technique available to doctors for the visualization of the body without surgery. It is generally noninvasive, except when used in methods such as angiography where a radiopaque substance is injected into the bloodstream to highlight the circulation in any part of the body [20]. However, X-rays have ionizing effects on the body and therefore should not be repeatedly used. X-rays were first discovered by Wilhelm Röntgen in 1895. The technique involves having a film or screen containing a radiation-sensitive material exposed to the x-rays transmitted through a region of the body. The developed film or excited phosphorous screen exhibits a geometric pattern produced by the structures in the beam path [20]. X-ray imaging is limited as the signal can be reduced due to the scattering of a large percentage of radiation from the body, and much detail is lost in the radiographic process with the superposition of 3D structural information onto a 2D surface. Therefore the 3D nature of bones, muscles, ligaments and vessels are all hard to capture on X-ray. The use of X-rays is usually limited to scanning bone. Fig. 1.1 shows an angiography of the left ventricle.

Figure 1.1: A left ventricle angiography (courtesy of Fu Jin (fjin@engmail.uwaterloo.ca))

1.1.2 Ultrasound Image

An ultrasound image is based on using a propagating ultrasonic wave that partially reflects at the interface between different tissues [23]. The reflections are measured as a function of time, and the position of the tissue can thus be obtained if the velocity of the wave in the medium is known. As the wave propagates through the body, diffraction, refraction, dispersion, and scattering occurs and can affect the image quality. Speckle noise patterns due to scatter are common, but are important to help the user distinguish between different tissues. There are also possible artifacts due to reverberations when waves are reflected back and forth between the transducer and the tissue. Ultrasound is commonly used for soft tissues, fluids, and small calcifications that are preferably close to the patient's body surface and not hidden by bony structures.

1.1.3 Magnetic Resonance Image

Magnetic resonance imaging was developed in the early 1970s and has become a versatile and clinically useful diagnostic imaging modality [20]. In contrast to X-ray and CT, MRI is a noninvasive imaging technology that does not use ionizing radiation. It is based on perturbing magnetic fields with radiowaves. In MRI, hydrogen nuclei (protons) are imaged due to their strong magnetic moment and prevalence in the soft tissues of the body (water molecules). The signal being measured can be controlled through modulation of the magnetic field and radiofrequency pulse sequences used to alter the spins of protons in the structure being imaged.

MRI best captures human soft tissue anatomy, as it is able to provide high contrast between soft tissues. In addition, its signal is not disturbed by bone. It is therefore used often to find pathological changes such as tumours, haemorrhages and inflammations, and the central nervous system where there are many bones [12].

Unlike many other medical imaging modalities, the contrast in an MR image depends strongly upon the way the image is acquired. By altering RF and gradient pulses, and choosing relaxation timings, it is possible to highlight different components in the object being imaged and produce high contrast images. These two features facilitate segmentation. On the other hand, ideal imaging conditions are never realized in practice. Image information is degraded considerably by electronic noise, the bias field (intensity inhomogeneities in the RF field) and the partial-volume effect (multiple tissue class occupation within a voxel), all of which cause classes to overlap in the image intensity histogram. Moreover, MR images are not always high-contrast. Many T2-weighted and proton density images have low contrast between gray matter and white matter [29]. Fig. 1.2 shows an MRI of the brain.

1.2 Segmentation Methods

Popular image segmentation methods include thresholding, region-growing, and active contours. Thresholding is the technique where an image is segmented based on peaks and valleys of the

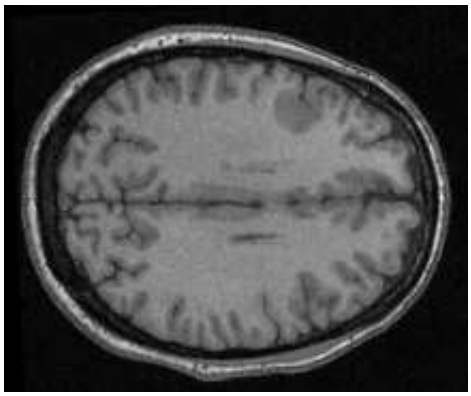


Figure 1.2: A brain MRI (courtesy of Merge Efilm)

histogram of a certain image feature, the simplest being pixel intensity. The method uses the valleys as the thresholding limits for distinguishing the regions. The method is appropriate for images with homogeneous regions but problematic with image flaws such as noise or broken edges.

Region growing divides an image into regions by grouping adjacent pixels with similar gray values, resulting in boundaries between contrasting regions. Region growing techniques begin with seed points, which grow by iteratively merging adjacent pixels with similar gray values. Problems with the region growing technique are similar to thresholding, where noise and unclear edges affect the performance greatly [20].

The method of active contours, also commonly known as snakes, on the other hand, involves minimizing a so-called energy function which is based on certain properties of the desired object boundary, for example the smoothness of the boundary curve and local gradient of the image. Starting with an initial curve, sometimes in a seed form or user-defined, the curve expands and contracts until the minimum energy function is found. This way, the resulting segmentation avoids yielding broken boundaries even with the presence of noise and non-ideal edges. However, there is the possibility of getting caught in local minima, influenced by the initial curve [20]. In addition, since the curve flow is modeled by differential equations, iterative methods are used to find the minimum and the computation speed is usually slow.

1.3 Motivation

As mentioned in the beginning of this chapter, image segmentation is important in various applications. Most segmentations are still done by radiologists, however, as human vision is still the most reliable and intelligent way of segmenting objects of interest. With high volume, high resolution images such as MRI, the radiologist's job is tedious. The demand to automate the segmentation process is great and fully-automatic algorithms are much desired. Meanwhile, although there exists a wide range of methods such as ones described in sec. 1.2 and approaches devoted to medical image segmentation, the problem remains a challenging one, as medical images are plagued with poor contrast, noise, poorly-defined boundaries and complications such as overlapping tissues that even the human eye may have difficulty in distinguishing. Furthermore, while the imaging technology may contribute to difficult image artifacts, as discussed in section 1.1, human factors such as patient's movement while doing a MRI scan also cause inconsistency in image quality, making a fully automatic segmentation method difficult. The incorporation of user interaction is common in commercial applications (e.g. 3D-Doctor^{TM}), and there is also much research in the incorporation of prior information such as shape [24] and texture [18].

This work examines closely a particular segmentation method called the geometric active contour (proposed by [4], [13]), and attempts to incorporate multi-resolution methods to help tackle the afore-cited problems with medical image segmentation.

1.4 Thesis Overview

The objective of this thesis is to examine the effectiveness of the geometric active contour method in the segmentation of medical images and how the incorporation of multi-resolution methods improve the process. Chapter two describes the active contour method. Chapter three shows results from an implementation of a geometric active contour. Chapter four discusses multi-resolution methods and their application to active contour methods. Chapter five describes

the proposed approach and analyzes the results from implementing the method. Chapter six summarizes and concludes the study.

Chapter 2

Active Contours and Level Sets

In recent years, techniques of active contours and curve evolution have been widely investigated and applied to the image segmentation problem. Compared to other methods such as thresholding and edge-based methods, the active contour model has the advantage of being less sensitive to blurred edges and also avoiding broken contour lines. In general, active contour models are based on deforming an initial contour C towards the boundary of the object to be detected, through minimizing a functional designed such that its minimum is obtained at the boundary of the object. Energy minimization involving components controlling the smoothness of the curve and one for pulling the curve closer to the boundary is a common technique. There are two main types of active contour models: parametric or geometric. Parametric, or "explicit", active contours are classical snake models first introduced by Kass et al. [9], where the curve is defined explicitly by the curves points, and moves the points according to the energy function. The geometric active contours, or "implicit" active contours, on the other hand, implicitly represent the curve by embedding the curve in a higher dimension function, and then evolve this function instead. Both are discussed further in the next sections.

2.1 Snakes

In the classical energy based snakes approach, first introduced by Kass et al. [9], the curve C is associated with an energy given by Eq. 2.1, where p parametrizes the curve $0 \leq p \leq 1$, α, β, and λ are real positive constants. The first two terms control the smoothness of the contours to be detected, where α controls the "tension", β controls the "rigidity", while the third term is responsible for attracting the contour towards the object in the image. In this approach, the curve is parameterized with points (x_i, y_i) and arclength. This method of parameterizing the curve is known as the Langrangian approach (common in fluid mechanics problems), which focuses attention on material particles as they move through the flow [17]. Although effective in giving a continuous boundary, this approach cannot directly deal with changes in topology. The curve points may "tangle" and "collide" as the curve evolves. It is thus crucial that prior knowledge is obtained and a reasonable initial curve be set such that the algorithm gives meaningful results.

$$E(C) = \alpha \int_0^1 |C'(p)|^2 dp + \beta \int_0^1 |C''(p)|^2 dp - \lambda \int_0^1 |\nabla I(C'(p))| dp \tag{2.1}$$

2.2 Geometric Active Contours

Caselles et al. [4] and Malladi et al. [13], at around the same time, independently further enhanced the idea of snakes by developing a curve evolution flow which can better accommodate the object geometry and the changing topology as the curve evolves. The basic idea stems from Euclidean curve shortening and the use of level sets when evolving the curve. First, let C=C(p,t) be a smooth family of closed curves where t parameterizes the family and p the given curve, $0 \leq p \leq 1$. consider the length functional [4]:

$$L(t) = \int_0^1 |\frac{\partial C}{\partial p}| dp$$

Differentiating using integration by parts (where $\langle \vec{A}, \vec{B} \rangle$ is the dot product of $\vec{A}, \vec{B}$),

$$L'(t) = \int_0^1 \frac{\langle \frac{\partial C}{\partial p}, \frac{\partial^2 C}{\partial p \partial t} \rangle}{|\frac{\partial C}{\partial p}|} dp$$
$$= -\int_0^1 \langle \frac{\partial C}{\partial t}, \frac{1}{|\frac{\partial C}{\partial p}|} \frac{\partial}{\partial p} \left[\frac{\frac{\partial C}{\partial p}}{|\frac{\partial C}{\partial p}|} \right] |\frac{\partial C}{\partial p}| \rangle$$

Since

$$|\frac{\partial C}{\partial p}| dp = ds$$

where ds is the arc-length, then

$$L'(t) = -\int_0^{L(t)} \langle \frac{\partial C}{\partial t}, \kappa \vec{N} \rangle ds$$

since $\frac{\partial \vec{T}}{\partial s} = \kappa \vec{N}$, where κ is the curvature, $\vec{T}$ is the tangent, and $\vec{N}$ is the inward normal vector. L(t) is decreasing most rapidly when $\frac{\partial C}{\partial t} = \kappa \vec{N}$.

In the work by Caselles et al. [4], this length-minimizing curve flow is modified through the multiplication of an edge-detector function g(I) (g approaches 0 if it is an edge):

$$L(t) = \int_0^1 g(I) |\frac{\partial C}{\partial p}| dp$$

Through the incorporation of the edge-detector, the flow moves to minimize the curve length but stops at an edge. Similar to the earlier curve flow derivation, the curve evolution equation becomes (derivation can be found in Appendix B in [4]):

$$\frac{\partial C(t)}{\partial t} = g(I) \kappa \vec{N} - (\nabla g \cdot \vec{N}) \vec{N} \tag{2.2}$$

The level-sets approach, first proposed by Osher and Sethian [16] is then incorporated into the model to make it topology free. It assumes that the curve C is a level-set of a function ϕ, so C coincides with the set of points ϕ=constant, which is usually set to 0, meaning that C is

the 0-level-set of u. By embedding the evolution of C in that of u, topological changes of C(t) are handled automatically. As was shown in the Appendix C in [4], if the planar curve C evolves according to

$$\frac{\partial C(t)}{\partial t} = \beta \vec{N}$$

for a given function β, then the embedding function ϕ should deform according to

$$\frac{\partial \phi}{\partial t} = \beta |\nabla \phi|$$

Therefore the geodesic active contour model becomes:

$$\frac{\partial \phi}{\partial t} = g(I)(\kappa + c)|\nabla \phi| + \nabla g \cdot \nabla \phi \tag{2.3}$$

with a parameter c added to help speed up the convergence (note $\vec{N} = -\frac{\nabla \phi}{|\nabla \phi|}$).

An example of the level-set representation is shown in Fig. 2.1. The figure shows the level set expanding outward at a uniform speed in all directions, and as time passes (t=0 to t=2) the level set function (the right column) grows and the corresponding zero-level set curve (left column) changes to a circle of bigger radius.

The object boundary is then given by the zero level-set of the steady state (ϕ_t=0) of this flow. This approach is Eulerian, which focuses on a fixed point in space as time proceeds [17], capturing the interface through the implicit function ϕ, as opposed to tracking the movement of points as in the Langrangian approach with snakes.

2.2.1 Signed Distance Function

The higher-dimensional function ϕ(x,y,t) is usually set as a signed distance function:

$$\phi(x, y, 0) = \pm d(x, y)$$

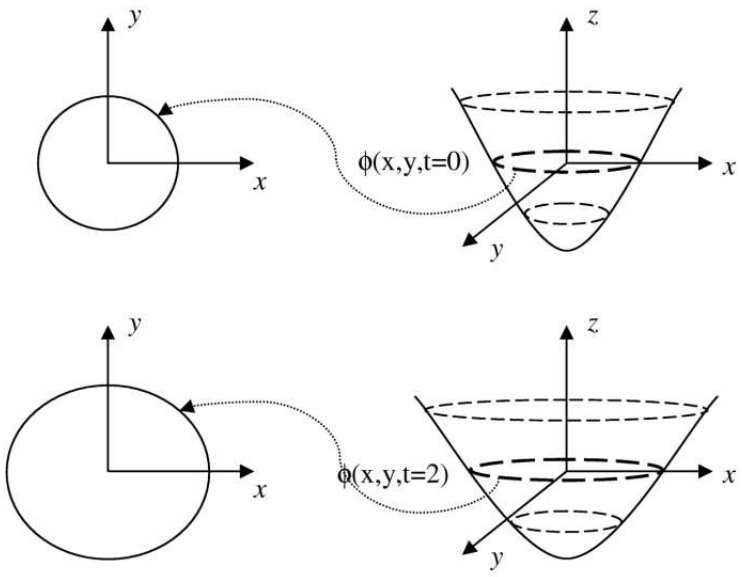


Figure 2.1: Level-set representation: a circle represented by a level set function and expanded at a uniform speed (based on [19])

where d(x,y) is the distance from (x,y) to the closest point on the curve C, and is positive if (x,y) is outside the curve, negative if (x,y) is inside the curve. Fig. 2.2 illustrates the 2-dimensional signed distance function of the set of 1-dimensional points 1,-1. The use of the signed distance function simplifies the numerical scheme to be used, as will be further explained later in section 2.3. It also ensures a smooth surface for the evolution, avoiding potential convergence instabilities due to numerical approximations.

2.3 Implementation of Geometric Active Contours

Implementing eq. 2.3 requires careful consideration of numerical schemes to ensure stable convergence [13]. Furthermore, as level set function evolves, the function does not remain a signed

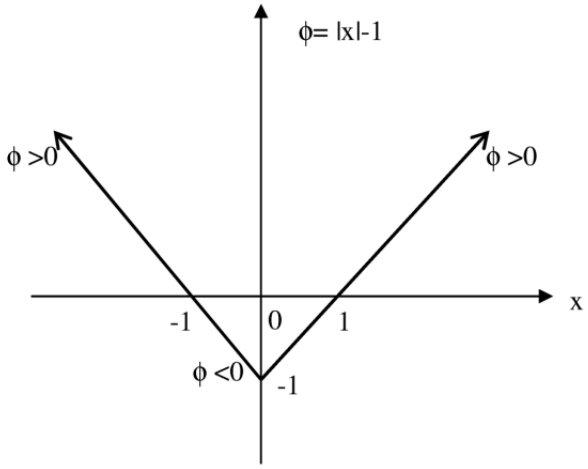


Figure 2.2: Signed distance function (based on [15])

distance function and therefore periodical re-initialization is needed to keep the function as a signed distance function. Also, because of the considerable computation involved, various algorithms are introduced to help increase the speed. These implementation aspects are discussed further in this section.

2.3.1 Numerical Schemes

First of all, the proper numerical methods are needed to ensure reasonable approximations for the discretization of the partial differential equation. To discretize and solve the equation numerically, finite difference approximations are needed. First order finite difference approximations of u(x,y) can be derived from Taylor series expansion. Central difference, forward difference, and backward difference approximations are shown in eqs. 2.4, 2.5, 2.6 respectively.

$$D_x^0 u = \frac{(u_{i+1,j} - u_{i-1,j})}{2\Delta x}, D_y^0 u = \frac{(u_{i,j+1} - u_{i,j-1})}{2\Delta y} \tag{2.4}$$

$$D_x^+ u = \frac{u_{i+1,j} - u_{i,j}}{\Delta x}, D_y^+ u = \frac{u_{i,j+1} - u_{i,j}}{\Delta y} \tag{2.5}$$

$$D_x^- u = \frac{u_{i,j} - u_{i-1,j}}{\Delta x}, D_y^- u = \frac{u_{i,j} - u_{i-1,j-1}}{\Delta y} \tag{2.6}$$

To understand what appropriate numerical methods should be used, the problem is broken down into three cases. Consider the general equation:

$$u_t + \vec{V} \cdot \nabla u = 0$$

where $\vec{V}$ is a speed term that moves the surface u. Consider the three cases where (1) $\vec{V}$ is an externally generated speed function $\vec{V}(\vec{x}, t)$, or (2) self-generated velocity field $\vec{V}$ that depends directly on the level set function u, or (3) a constant velocity field. Most of the material in this section is taken from the chapters 1-7 in [15].

Externally Generated Velocity Field

First consider the simple first order partial differential wave equation, eq. 2.7, in one dimension, where u=u(x,t), and a(x) is a given speed term that determines how fast the wave or each point on the surface moves.

$$\frac{\partial u}{\partial t} + a(x)\frac{\partial u}{\partial x} = 0 \tag{2.7}$$

Let $u_t = \frac{\partial u}{\partial t}$. Since the equation is evolved forward in time, a simple first-order forward difference can be used for u_t, as seen in eq. 2.8, where n indicates the iterations in time.

$$\frac{u^{n+1} - u^n}{\Delta t} + a^n u_x^n = 0 \tag{2.8}$$

The sign of a^n indicates whether the values of u are moving to the right or left. At grid point x_i, consider if $a_i > 0$, the values of u are moving from left to right. To determine what value of u

will land on the point x_i at the end of a time step, one should look to the left of x_i. Therefore, if $a_i > 0$, use backward difference for approximation and otherwise, use forward differencing. This scheme of choosing the approximation to the spatial derivatives based on the sign of a is known as upwind differencing, as seen in eq. 2.9.

$$u_x = \max(a, 0)D_x^- u + \min(a, 0)D_x^+ u \tag{2.9}$$

Note that although u_x can be approximated with central differencing, which is more accurate (second-order accurate), simple central differencing is unstable with forward Euler time discretization and the usual CFL conditions with $\Delta t \sim \Delta x$, potentially leading to unwanted oscillations [25].

Now, consider the equation of the form

$$u_t + H(\nabla u) = 0 \tag{2.10}$$

which is known as the general Hamilton-Jacobi equation where H can be a function of both space and time, and is known as the "Hamiltonian". Eq. 2.7 is an example of $H(\nabla u) = \vec{V} \cdot \nabla u$ in multi-dimension form. Note that Hamilton-Jacobi equations depend on (at most) the first derivatives of u, and these equations are hyperbolic[1]. To estimate the numerical scheme, hyperbolic conservation laws are used. For a convex function[2] F where the resulting Hamiltonian $H(\nabla u) = F|\nabla u|$ is also convex, $H(\nabla u)$ can be approximated as follows:

$$H_{convex}(\nabla u) = \max(F, 0)\nabla^+ + \min(F, 0)\nabla^- \tag{2.11}$$

where

[1]Eq. 2.13 introduced in the next section depends on the second derivatives of u, is parabolic, and is not an example of a Hamilton-Jacobi equation.

[2]In N dimensions, a function $F(x_1, ..., x_N)$ is convex if $F(\lambda p + (1-\lambda)q) \leq \lambda F(p) + (1-\lambda)F(q)$ for all $0 \leq \lambda \leq 1$, p, q $\in \mathbb{R}^N$. Equivalently, F is convex if $\frac{\partial^2 F}{\partial x_i \partial x_j} \geq 0$ for all i,j=1,2,....,N.

$$\nabla^+ = (\max(D_x^- u, 0)^2 + \min(D_x^+ u, 0)^2 + \max(D_y^- u, 0)^2 + \min(D_y^+ u, 0)^2)^{\frac{1}{2}}$$

$$\nabla^- = (\max(D_x^+ u, 0)^2 + \min(D_x^- u, 0)^2 + \max(D_y^+ u, 0)^2 + \min(D_y^- u, 0)^2)^{\frac{1}{2}}$$

If F is a non-convex function, then the following scheme is used instead:

$$H(\nabla u) = H_{convex}(\nabla u) - \frac{1}{2}\alpha_u(D_x^+ u - D_x^- u) - \frac{1}{2}\alpha_v(D_y^+ u - D_y^- u) \tag{2.12}$$

where

$$\alpha_u = \max_{i,j} |D_x^0 u|$$

$$\alpha_v = \max_{i,j} |D_y^0 u|$$

Self-generated Velocity Field

Instead of a speed term (a) that varies spatially only, consider now a speed term that depends directly on the level set function u. The equation changes from being a linear partial differential equation to a non-linear one. In particular, consider the use of a speed term based on the surface curvature, which is closer to the equation of interest (eq. 2.3). Eq. 2.13 shows such an equation, where u is n-dimensional, κ is the curvature of u, and ∇u is its gradient.

$$u_t + b\kappa|\nabla u| = 0 \tag{2.13}$$

Compared with eq. 2.9, the term $b\kappa|\nabla u|$ is parabolic and cannot be discretized with an upwind approach [15]. When u is a signed distance function, $|\nabla u| = 1$, $\kappa = \Delta u$, eq. 2.13 becomes the heat equation, a basic equation of the parabolic model,

$$u_t + b\Delta u = 0 \tag{2.14}$$

where u is the temperature and b is the thermal conductivity. Such parabolic equations need to be discretized using central differencing since the domain of dependence includes information from all spatial directions, as opposed to hyperbolic equations like eq. 2.7 [15]. Therefore, to discretize eq. 2.13, κ is discretized using a second-order central differencing scheme (in two dimensions):

$$\kappa = \frac{u_x u_y^2 - 2u_y u_x u_{xy} + u_{yy} u_x^2}{|\nabla u|^3} \tag{2.15}$$

For eq. 2.14, when $|\nabla u| = 1$, Δu, the Laplacian of u, is given by

$$\Delta u = u_{xx} + u_{yy} \tag{2.16}$$

and can be discretized using second-order accurate central differencing formula in each direction:

$$\frac{\partial^2 u}{\partial x^2} \approx D_x^+ D_x^- u = D_x^- D_x^+ u = \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} \tag{2.17}$$

Therefore having u as a signed distance function simplifies the discretization of $b\kappa|\nabla u|$ by substituting it with $b\Delta u$.

Constant Velocity Field

Consider eq. 2.18

$$u_t + a|\nabla u| = 0 \tag{2.18}$$

When u is a signed distance function, the equation reduces to $u_t = -a$ and if u^n is initially a signed distance function, it stays a distance function for all time. Using forward time discretization, the solution of eq. 2.18 is straightforward:

$$u = -a_0 t + u^0 \tag{2.19}$$

Therefore, using u as a signed distance function greatly simplifies the numerical scheme for the case with a constant velocity field.

The CFL Condition

To ensure that small errors in the approximation are not amplified as the solution is evolving in time, in other words, that the numerical scheme is stable, the Courant-Friedreichs-Lewy condition (CFL condition) can be used [15]. The CFL condition asserts that the numerical solution should propagate as fast as the physical solution, and it does so by restricting the time step:

$$\delta t < \frac{\Delta x}{\max\{|u|\}} \tag{2.20}$$

where $\max\{|u|\}$ is chosen to be the largest value of $|u|$ over the entire Cartesian grid.

Summary

Since both the first and last term are externally generated velocity fields while the second term is curvature-dependent, eq. 2.3 is implemented as follows:

$$\begin{aligned} u^{n+1} = u^n + \delta t \Big\{ & \underbrace{(\max(c^n,0)\nabla^+ + \min(c^n,0)\nabla^-)}_{\text{upwind difference}} + \underbrace{\kappa\sqrt{(D_x^0)^2 + (D_y^0)^2}}_{\text{central difference}} \\ & - \underbrace{(\max(u^n,0)D_x^- + \min(u^n,0)D_x^+ + max(v^n,0)D_y^- + max(v^n,0)D_y^-)}_{\text{upwind difference}} \Big\} \end{aligned} \tag{2.21}$$

where $\nabla g = (u, v)$ is the gradient of the edge detector (u is the x-component and v the y-component).

2.3.2 Signed Distance Function and Re-initialization

As seen in section 2.3.1, having the level set function as a signed distance function simplifies numerical schemes, as in the cases of constant velocity field and curvature-dependent velocity field. Furthermore, the level set function can develop noisy features and steep gradients that are not amenable to finite-difference approximations. Keeping the level set function as a signed distance function helps it to stay smooth enough to approximate its spatial derivatives with some degree of accuracy.

Even if the level set function is set as a signed distance function at the beginning of the evolution, it generally drifts away and does not stay as signed distance. Therefore periodic re-initialization of the level set function as a signed distance function is needed to ensure smooth evolution. How frequent the re-initialization is required depends on how sensitive the particular numerical approach is to how accurately the level set function approximates a signed distance function.

To set up the level set function as a signed distance function, the straightforward approach would be to determine the distance to the closest curve point for every point in the set. A speedier approach uses the idea of crossing times which can help locate and discretize the interface (at 0 level-set) [15]. The trick is to move the interface in the normal direction with speed term 1. For each point, the time when the interface crosses over it is when the value at the point changes from negative to positive or vice versa. Since the speed equals 1 and the level set is moved normal to itself, this time, the so-called *crossing time*, is equal to the distance. For points inside the interface, use speed= -1; for points outside the interface, use speed= 1. The advantage of the crossing time method is that the interface does not have to be located beforehand.

Another approach re-initializes by solving the partial differential equation 2.22 [15]. Upon convergence of the equation, $|\nabla\phi| = 1$, thus establishing a signed distance function. This approach is fast but is not as accurate, compared to the above two approaches as the sign function does not provide very accurate information for pixels closed to the front. If ϕ is not smooth or ϕ is much

steeper on one side of the interface than the other, it may cause the interface to move incorrectly from the starting position. More functions that may provide a better estimate are discussed in [15].

$$\frac{\partial \phi}{\partial t} = sign(\phi)(1 - |\nabla \phi|) \tag{2.22}$$

2.3.3 Front Propagation Algorithms

While a straight-forward implementation of the geometric active contour is to update the level set function of all pixels in the image at each iteration, it is computationally expensive. To help reduce the computation time, different algorithms were proposed, using different properties of the level set function. Initially proposed in [6], and further analyzed in [1], the narrow band approach evaluates the level set function only for a band of pixels close to the 0 level-set [19]. Fast march, introduced in [21], on the other hand, simplifies the level set function by assuming the speed function is always positive or negative and thus simplifies the computation needed. These two methods are discussed in more detail in this section.

Narrow Band

Since the front propagates smoothly, it cannot propagate too fast and therefore only the values of pixels close to the front changes at each iteration. In other words, during evolution, only a band of pixels around the front, both outside and inside the front, need to be considered, as seen in fig. 2.3. Checks need to be made at each iteration to make sure that the front is evolving within the band. When the front is close to the edge of the band, the band is re-initiated by establishing a new band around the current front and its correct signed distance values [19].

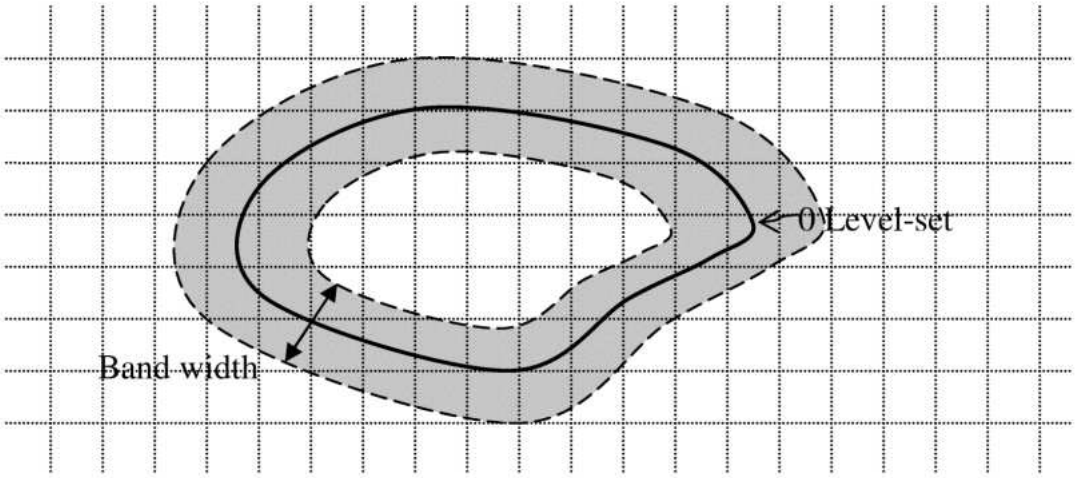


Figure 2.3: The use of narrow band for front propagation (based on [19])

Fast March

Consider when the speed term is either always positive or negative, and let T(x,y) be the time at which the curves cross the point (x,y). Then, given

$$u_t + F(x, y, z)|\nabla| \tag{2.23}$$

the following can be obtained

$$|\nabla T|F = 1 \tag{2.24}$$

which says that the gradient of arrival time surface is inversely proportional to the speed of the front.

As discussed in sec 2.3.1, in the case of an externally generated speed term, the level set function is discretized using eq. 2.11. Using the same numerical scheme, eq. 2.24 becomes:

$$\frac{1}{F^2} = \max(D_x^- T, 0)^2 + \min(D_x^+ T, 0)^2 + \max(D_y^- T, 0)^2 + \min(D_y^+ T, 0)^2 \tag{2.25}$$

One can solve this equation iteratively, but it would take considerable computation. Since for eq. 2.24, the front only propagates in one direction, the solution can be obtained using an outward propagation scheme from the smallest T value[3]. The algorithm first finds the front point with the smallest T value and then considers moving the front to its neighboring points. The algorithm splits pixels into three categories: (1) **alive**, which are points already reached by the front and their crossing times are known; (2) **active**, which are points that are candidates to be reached by the front in the current iteration; and (3) **far away**, which are points that have not been considered and their crossing times are unknown. Then the algorithm, as described succinctly in [19], goes as follows:

1. Initialize
   (a) Front pixels constitute the set of Alive pixels that are assigned to zero crossing time [T=0]
   (b) For each Alive pixel, its neighborhood pixels are examined. If they are not Alive pixels, label them as Active pixels with $T_(i,j) = \frac{1}{F_(i,j)}$
   (c) Rest of the pixels are labelled as Far Away points with $T_(i,j) = \infty$
2. Marching Forward
   (a) Pixel with the smallest value of T within the Active set is selected. This pixel is removed from the Active set and is labelled as Alive
   (b) Neighboring pixels of the selected pixel are examined. If they are labelled as Far Away, they will become Active. Update the T value of this and the existing Active set of pixels according to eq. 2.25, selecting the largest possible solution to the quadratic equation.

[3]This idea is further explained and proven in [21].

(c) Repeat Marching Forward if active set is not empty

Although the fast march algorithm has lower computational cost, the main limitation remains its assumption that the speed term is always positive or negative. This limits the choices of the speed term, which, for instance, cannot be a curvature-dependent speed function. However, it can still serve very well as a rough but fast initial attempt to boundary capturing, if the initial front is set up as a seed inside the object. Then it is similar to region growing, where the front propagates from a seed, outward until it hits a boundary. Also, it can be used for re-initialization using the crossing time idea, as described in section 2.3.2, since the crossing time idea uses a constant speed function of F=1 and F=-1.

2.4 Summary

The ideas of snakes and geometric active contour help solve the segmentation problem by attempting to capture boundaries through a continuous curve. Geometric active contour overcomes previous problems faced by snakes by using an Eulerian formulation of the evolution function instead of a Langrangian approach, and is thus able to adapt to topology changes. Meanwhile, the implementation can be computationally intensive and the methods narrow band and fast marching are attempts to speed up the process and are introduced in this section.

The design of the energy function is crucial in the success of the boundary extraction and may depend on the specific applications. While earlier work on active contours mainly focuses on the use of edge as a feature in the energy function, recent works explore more use of statistics and more regional or global approaches, as opposed to local edge information. Edge-based methods have problems with poorly defined edges and noise, which may cause the evolution to be stuck at a local optimum or evolve past weak edges.

Chapter 3

Geodesic Active Contour Demonstration

To demonstrate how geometric active contours fair in segmentation, the geodesic active contour, proposed by Caselles et al. [4] is implemented and tested. The evolution function (eq. 3.1) is implemented, with parameters (c, β, ε), using numerical methods as described in section 2.3.1. All parameters are set to 1 unless otherwise specified. The edge detector g(I) used is shown in eq. 3.2, where k is a positive constant parameter (again, set to 1 unless otherwise specified). The algorithm is implemented in Java and tested on a Celeron 433MHz computer with 192MB RAM. Initial curve (a box) coordinates are given by the user.

$$\frac{\partial \phi}{\partial t} = g(\beta\kappa + c)|\nabla\phi| + \varepsilon \nabla g \cdot \nabla\phi \tag{3.1}$$

$$g = \frac{1}{1 + (\frac{\nabla I}{k})^2} \tag{3.2}$$

The narrowband algorithm is implemented to speed up the process. The level set function is re-initialized whenever $\nabla\phi > 2$ to keep the function smooth and remain as a signed distance

function. $\nabla\phi > 2$ is picked arbitrarily and is found to give reasonable results while not being too computationally exhausting. Note that re-initializing too often (such as setting $\nabla\phi > 1$) may cause the level set to remain stationary. Re-initialization is done through the third method introduced in sec. 2.3.2, using eq. 3.3 [25], which is a modified version of eq. 2.22. Since the narrow band algorithm is used, the level set function is also re-initialized when the front reaches the band boundary.

$$\frac{\partial\phi}{\partial t} = sign(\phi)\min(1,\phi)(1 - |\nabla\phi|) \tag{3.3}$$

The CFL condition as mentioned in sec 2.3.1 is maintained by ensuring that $\delta t < \frac{0.5}{\max\{|u|\}}$, and if not δt is adjusted to $\delta t = \frac{0.3}{\max\{|u|\}}$. Again, the constraints 0.5 and 0.3 are chosen arbitrarily to conservatively fulfil the CFL condition.

The segmentation stops when the level set equation reaches steady state. This is done by checking if there is change in the front length. When the number of front pixels remain the same for a specified number of iterations (set to 500 in most cases), the algorithm stops.

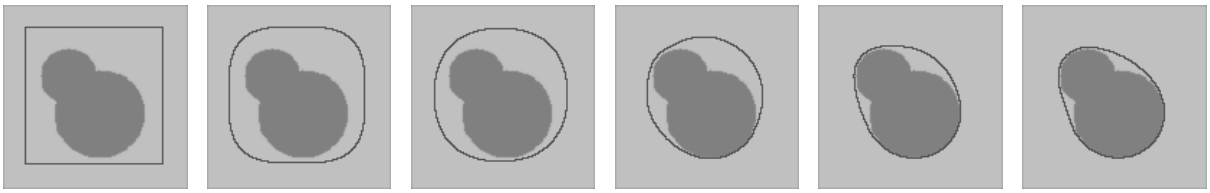
The nature of each term in the evolution equation is first demonstrated. Then further tests are shown to see performance of the algorithm under poorly defined edges and noise. The test images used in this section are combination of 128x128 synthetic images and some medical images.

Note that the computation time not only depends on the image size but also the frequency of re-initialization, which varies from image to image. This is because the frequency of re-initialization differs from image to image, depending on smoothness of the propagation, and the more frequent the re-initialization occurs, the more time it takes. It also varies from different initializations and different time intervals. In this work, the computation time is included in the figure captions for each example.

3.1 Level Set Equation

As shown in eq. 3.1, there are three main speed terms that play a role in advancing the front. First, let us consider the term $g(I)\kappa|\nabla\phi|$. On its own without g(I), the term $\kappa|\nabla\phi|$ is a Euclidean length minimizing term, thus keeping the evolving curve smooth. Multiplied by g(I), the speed term is adjusted to stop at the boundary. In the case of an ideal edge, the term goes to 0 and thus no movement, however large the curvature κ may be.

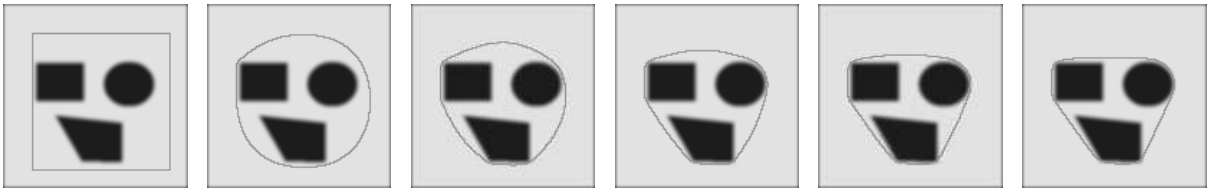
Fig. 3.1 and 3.2 demonstrate the evolution according to this term only. Although able to adhere to some boundary, convergence is slow, and the curvature constraint limits the curve to a minimal length, stopping the curve from reaching sharper corners and concave areas.



(a) (b) (c) (d) (e) (f)

Figure 3.1: Geodesic active contour with only the curvature term, a) initial, and after b) 5000, c) 10000, d) 20000, e) 30000 and f) 36000 iterations. (13 minutes)

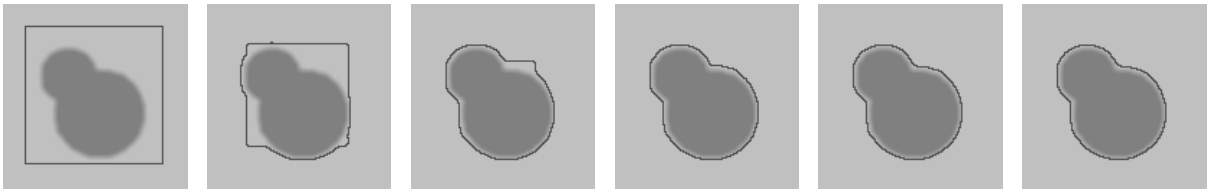
Now consider the constant advection term $cg(I)|\nabla\phi|$. As it was shown in Figs. 3.1, 3.2, when c=0 the convergence can be quite slow and this term is used to speed up the evolution. Fig. 3.3, 3.4 show how the constant advection term helps in pushing the curve closer to the boundary and giving a faster convergence as compared to Fig. 3.1, 3.2. The constant c can also be used to control the direction of curve flow. Since $|\nabla\phi|$ leads to an inward flow normal to the curve (remember that $\frac{\partial\phi}{\partial t} = \beta|\nabla\phi|$ is equivalent to $\frac{\partial C(t)}{\partial t} = \beta\vec{N}$, where $\vec{N}$ is the inward normal), to have an outward flow, c can be adjusted to a negative speed. Otherwise, unless the initial curve is close enough to the edge, the boundary would not be captured since the front never advances



(a) (b) (c) (d) (e) (f)

Figure 3.2: Geodesic active contour with only the curvature term, a) initial, and after b) 25000, c) 50000, d) 75000, e) 100000 and f) 125000 iterations. (14 minutes)

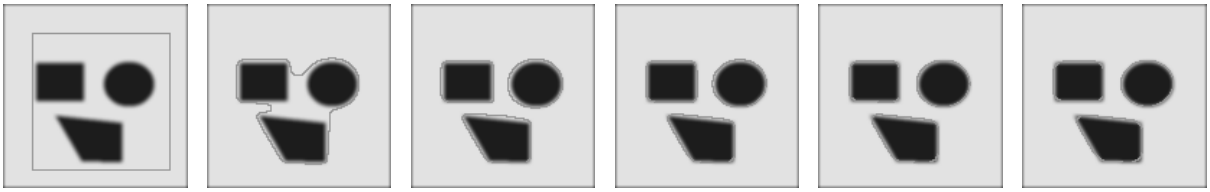
within the vicinity of the edge. This dependence on the locality of edges means that the initial curve should be set either completely inside or outside of the object of interest. The sign of c is then set accordingly to move either inward or outward. Fig. 3.5 shows an example of outward flow.



(a) (b) (c) (d) (e) (f)

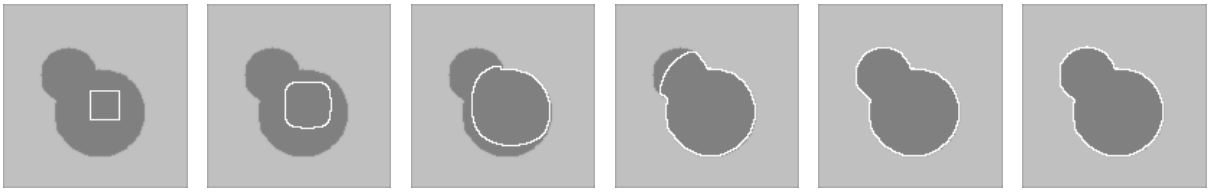
Figure 3.3: Geodesic active contour with both the advection and curvature term, a) initial, and after b) 2000, c) 4000, d) 5000, e) 6000 and f) 7000 iterations. (2 minutes)

The selection of c, as noted in several papers [22] [27], if chosen too large, can help with faster convergence but may cause overshooting of the edge, especially when an edge is non-ideal, e.g. blurred. This overshooting, highly likely when the edge is not strong, is prevented by the third term in the equation, which is introduced in [4]. The third term $\nabla g \cdot \nabla \phi$ (known as the doublet



(a) (b) (c) (d) (e) (f)

Figure 3.4: Geodesic active contour with both the advection and curvature term, a) initial, and after b) 5000, c) 10000, d) 15000, e) 200000 and f) 25000 iterations. (11 minutes)



(a) (b) (c) (d) (e) (f)

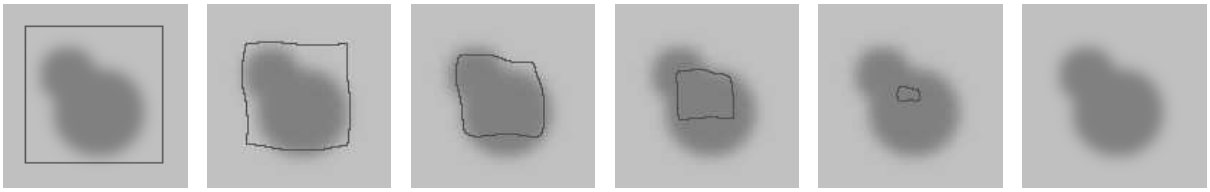
Figure 3.5: Geodesic active contour with both the advection and curvature term, outward motion, a) initial, and after b) 1000, c) 3000, d) 5000, e) 7000 and f) 11000 iterations. (3 minutes)

term), because of ∇g, keeps the front around the vicinity of the edge. Fig. 3.6, 3.7 shows how the third term helps locate the boundary in the case of a weak boundary.

Fig. 3.8, 3.9 show how the third term enables a closer and faster convergence, when compared to the results in fig. 3.3, 3.4.

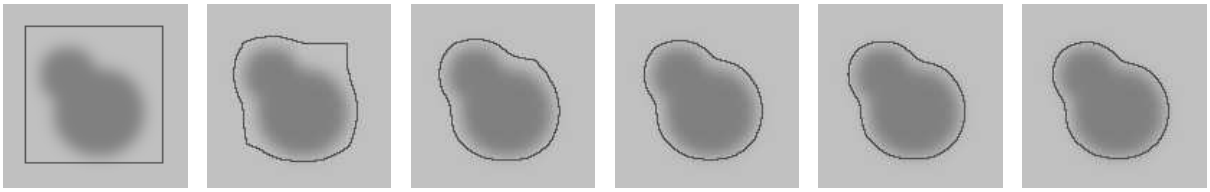
3.2 Contrast

Contrast of the image or edge affects the magnitude of the stopping force at the boundary. In cases of poor contrast, although the edge magnitude is smaller, due to the doublet term, a low-



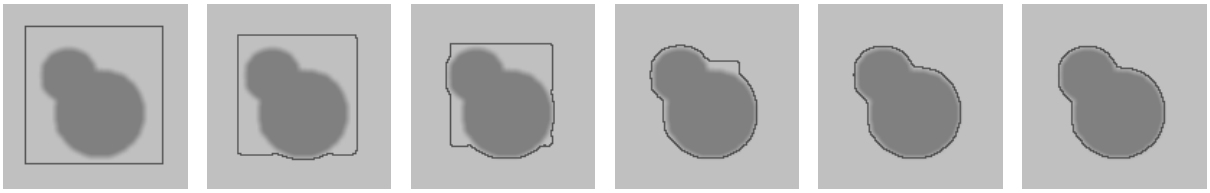
(a) (b) (c) (d) (e) (f)

Figure 3.6: Geodesic active contour with advection and curvature terms, a) initial, and after b) 2000, c) 4000, d) 6000, e) 8000 and f) 10000 iterations. (2 minutes)



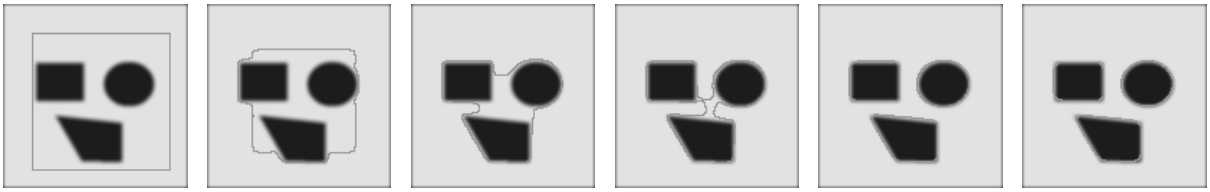
(a) (b) (c) (d) (e) (f)

Figure 3.7: Geodesic active contour with all terms, a) initial, and after b) 2000, c) 4000, d) 6000, e) 8000 and f) 10000 iterations. (2 minutes)



(a) (b) (c) (d) (e) (f)

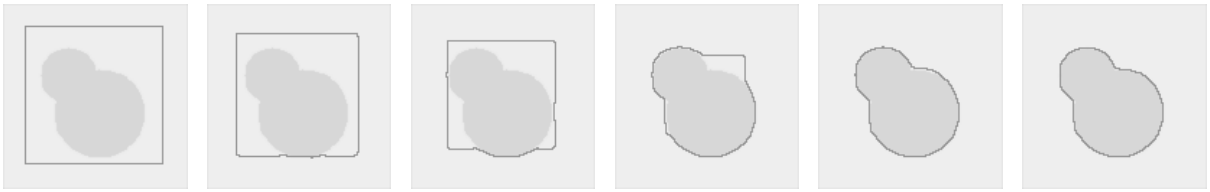
Figure 3.8: Geodesic active contour with all terms, a) initial, and after b) 1000, c) 2000, d) 4000, e) 6000 and f) 8000 iterations. (3 minutes)



(a) (b) (c) (d) (e) (f)

Figure 3.9: Geodesic active contour with all terms, a) initial, and after b) 2000, c) 5000, d) 8000, e) 11000 and f) 16000 iterations. (7 minutes)

contrast but clearly-defined boundary can still be captured. Fig. 3.10 shows a test image with low contrast.



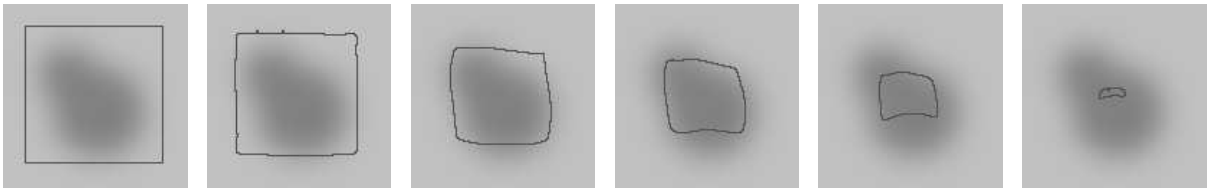
(a) (b) (c) (d) (e) (f)

Figure 3.10: Test image with low contrast, a) initial, and after b) 1000, c) 2000, d) 4000, e) 6000 and f) 8000 iterations. (2 minutes)

3.3 Weak Edges

Since eq. 3.1 relies heavily on edge information from the image, when an edge is less than ideal, e.g. blurred, or broken, the algorithm may "leak" beyond the supposed boundary. See fig 3.11.

Further example of edge leakages can be seen in fig. 3.15, fig. 3.21.

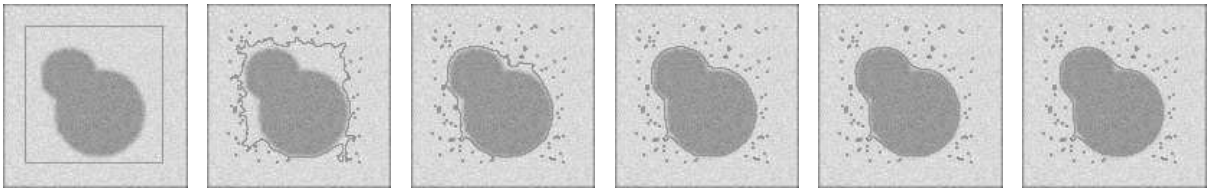


(a) (b) (c) (d) (e) (f)

Figure 3.11: Test image with blurred edge, a) initial, and after b) 1000, c) 3000, d) 5000, e) 7000 and f) 9000 iterations. (2 minutes)

3.4 Noise

Noise can greatly affect the segmentation results. Since the algorithm is edge-based and highly localized, noise can present local optimums where the curve evolution will get stuck. See fig 3.12.



(a) (b) (c) (d) (e) (f)

Figure 3.12: Test image with gaussian noise (mean=0.1, s.d.=0.005), a) initial, and after b) 5000, c) 10000, d) 15000, e) 20000 and f) 25000 iterations. (18 minutes)

3.5 Initialization

As mentioned in section 3.1, the algorithm is limited to a fixed direction of flow, according to the constant advection term c. This limits the placement of the initial curve to either being

completely exterior or interior to the real object boundaries.

3.6 Smoothing

It is generally helpful to smooth the image before applying geometric active contour. Most commonly Gaussian smoothing is applied. The amount of smoothing plays an important role in the success of the algorithm. Smoothing helps in eliminating noise that may hinder the propagation of the front. Fig. 3.13 show the different results from different degrees of Gaussian smoothing.

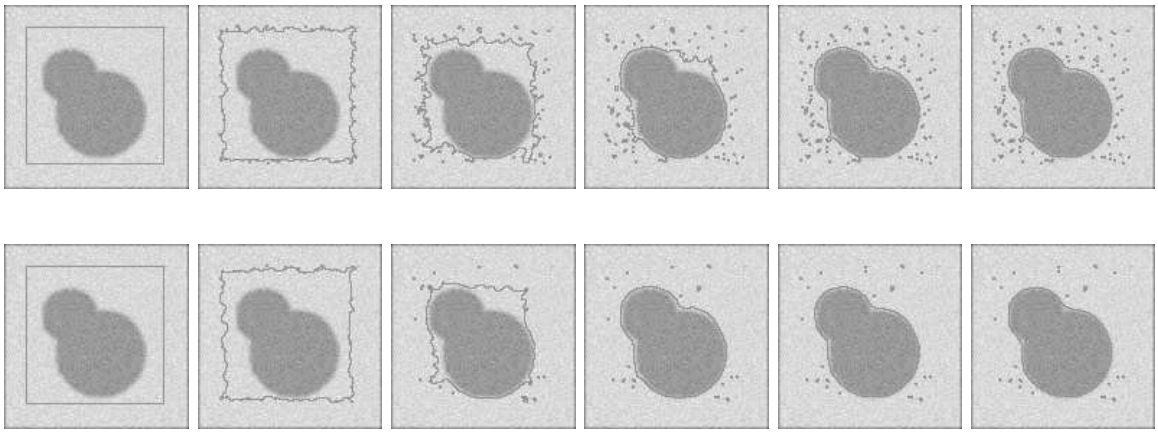
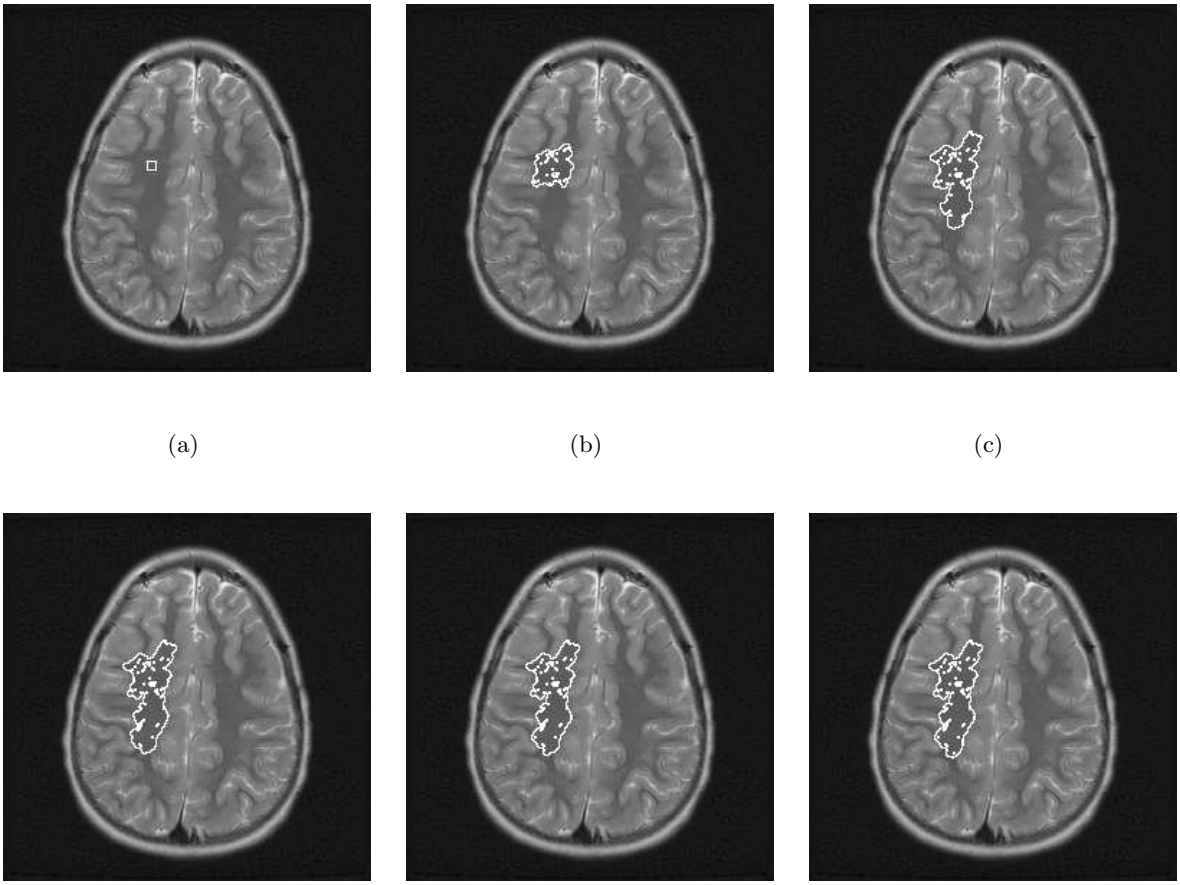


Figure 3.13: Noisy image with gaussian smoothing, first row: $\sigma = 1$; second row: $\sigma = 1.2$

Meanwhile, as the algorithm is quite sensitive to noise and weak edges, an adaptive smoothing filter such as the Lee filter may improve the results. Fig. 3.14, 3.15, 3.16 shows a test on the brain MRI image pre-processed with Gaussian smoothing and Lee filter smoothing. It can be seen that without Gaussian smoothing, the propagation is quite affected by noise. With Gaussian smoothing, it is slightly less prone to noise but suffers some edge leakage. The Lee filter avoids noise about the same degree as Gaussian but more successful in avoiding edge leakage.



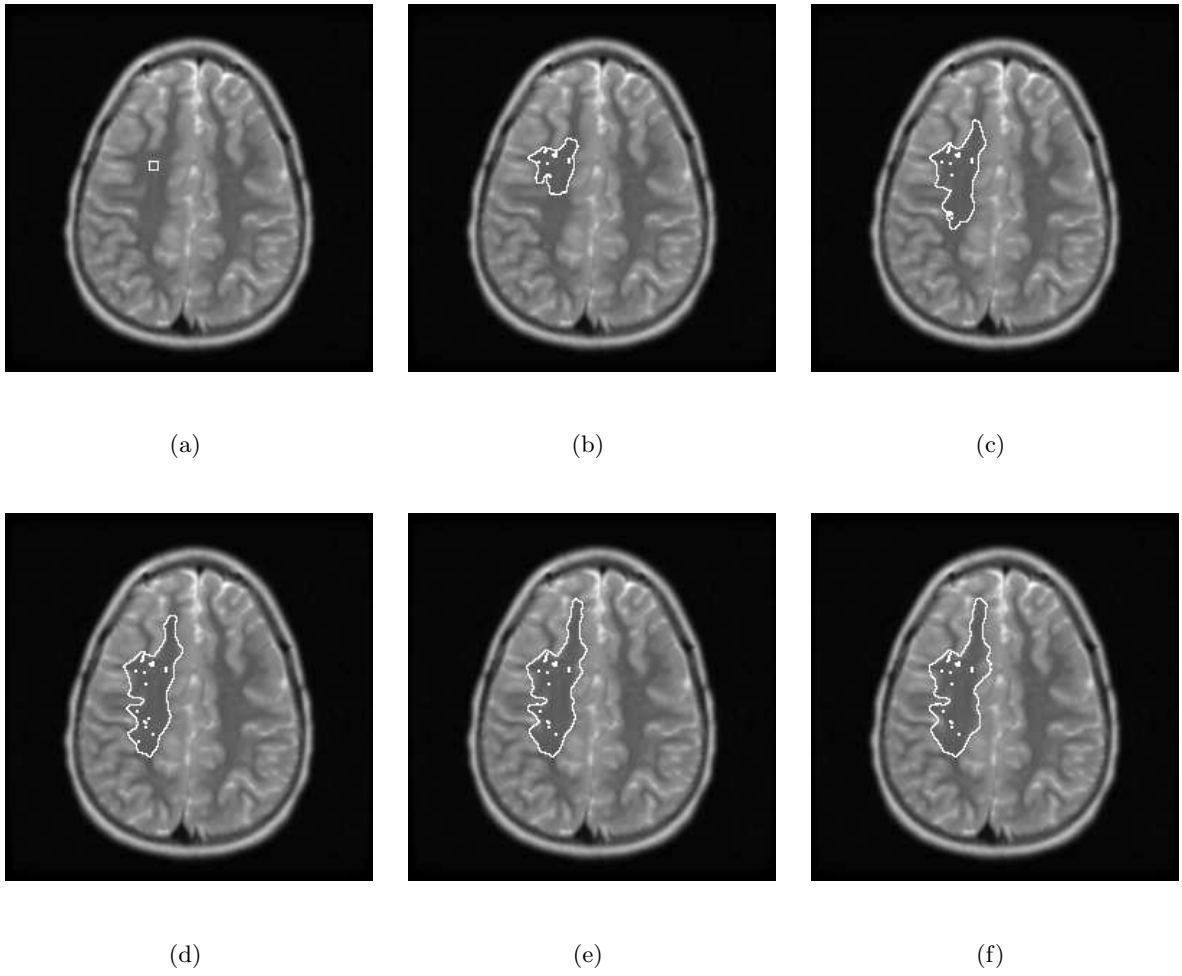
(a) (b) (c)

(d) (e) (f)

Figure 3.14: Test image (courtesy of Merge Efilm) preprocessed without smoothing , a) initial, and after b) 10000, c) 20000, d) 30000, e) 40000 and f) 50000 iterations.

3.7 Edge-detector

The most commonly used edge detector in PDE problems such as this, probably because of its simplicity, is eq. 3.2. The equation is, again:



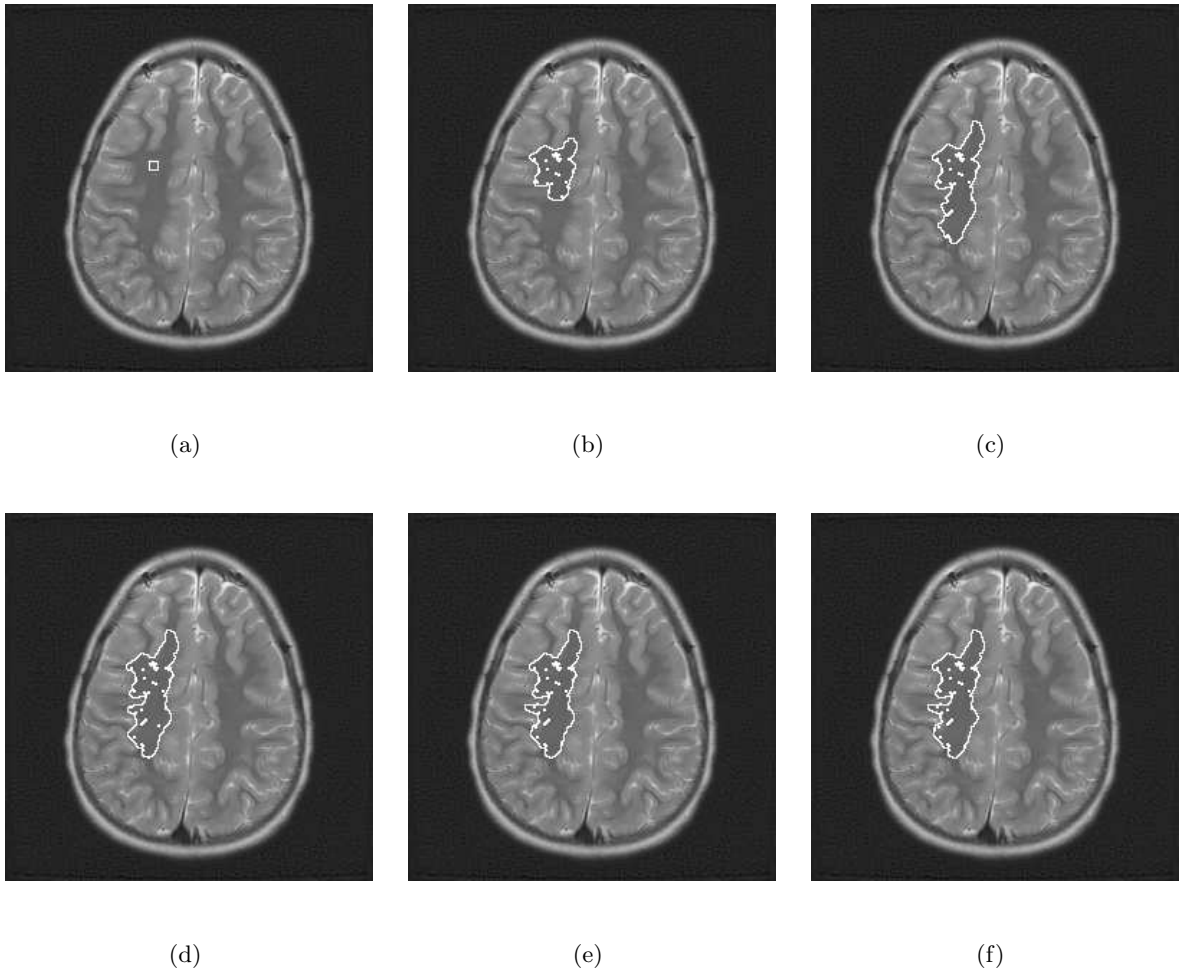
(a) (b) (c)

(d) (e) (f)

Figure 3.15: Test image (from Fig. 3.14) preprocessed with Gaussian smoothing (5x5, $\sigma = 1$), a) initial, and after b) 10000, c) 20000, d) 30000, e) 40000 and f) 50000 iterations.

$$g = \frac{1}{1 + (\frac{\nabla I}{k})^p}$$

where $p = 1, 2$, and k a positive constant for edge magnitude adjustment. $p = 2$ is most commonly used. It is a gradient-based edge-detector, where $g(I) \to 0$ as $\nabla I \to \infty$, and $g(I) \to 1$ as $\nabla I \to 0$. However, ideal edges rarely occur in real images and therefore, $g(I)$ is never quite completely 0.



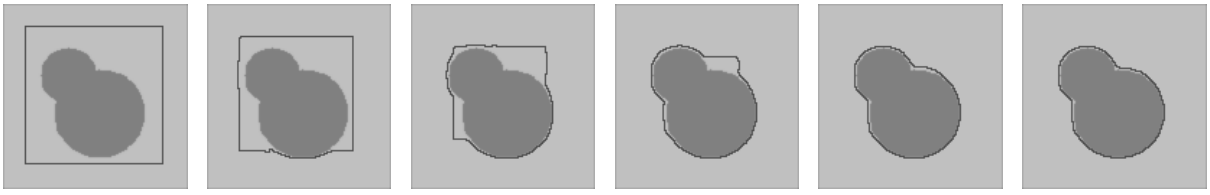
(a) (b) (c)

(d) (e) (f)

Figure 3.16: Test image (from Fig. 3.14) preprocessed with Lee filter of window 5x5, a) initial, and after b) 10000, c) 20000, d) 30000, e) 40000 and f) 50000 iterations.

In cases of weak edges, where the ∇I at the boundary not small enough, the algorithm can pass the boundary, even with the doublet term, as shown in previous tests. A better edge detector, less prone to noise, and more sensitive to real boundaries may improve the process. To verify, the gradient-based edge detector, the Sobel edge detector [8], and the Canny edge detector [3] were investigated. To ensure that the edge detector stays within the range of [0,1] and 0 indicating

edges, the Sobel edge is normalized and then inverted using $1 - g_{sobel}$. Similarly, the Canny edge detector is inverted using $1 - g_{canny}$. The Sobel detector is less sensitive to noise than the gradient, while Canny is a commonly-used robust edge detector. The results, however, do not show better propagation with the Canny edge detector. Fig. 3.17, 3.18, 3.19 show the results of the algorithm on a straightforward synthetic image with sharp edges using the three edge detectors. It is found that while Sobel gives the closest stop at the boundary, and gradient-based one gives good results, Canny leaks through. Upon inspection of the edge functions (see fig. 3.20), it is found that Canny has a steeper band of edge values, where bands of edge values of say 0.3 and 0.1 are close together, causing the algorithm to leak through. The Sobel edge map on the other hand has a wider band of edge values that are small enough (< 0.2). Adjusting the threshold of the Canny operator may help.

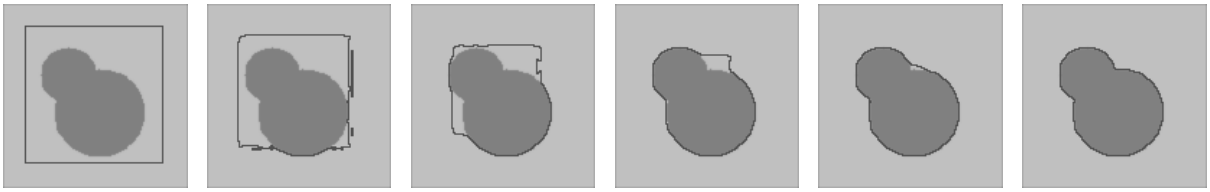


(a) (b) (c) (d) (e) (f)

Figure 3.17: Using gradient-based edge detector, a) initial, and after b) 1000, c) 2000, d) 3000, e) 4000 and f) 5000 iterations.

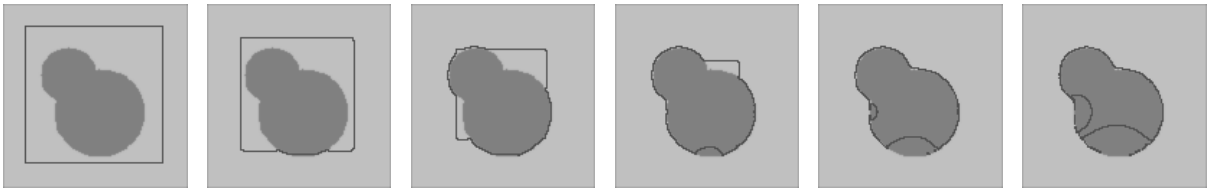
3.8 Further Tests

The parameters c, β, and ε (see eq. 3.1) are added to help negotiate the performance of the propagation. To see how the parameters affect each term in the level set equation and affect the outcome of the convergence, these parameters are varied and tested on images. β controls the curvature term, and c is the constant advection term which helps to push the propagation in a



(a) (b) (c) (d) (e) (f)

Figure 3.18: Using Sobel edge detector, a) initial, and after b) 1000, c) 2000, d) 3000, e) 4000 and f) 5000 iterations.

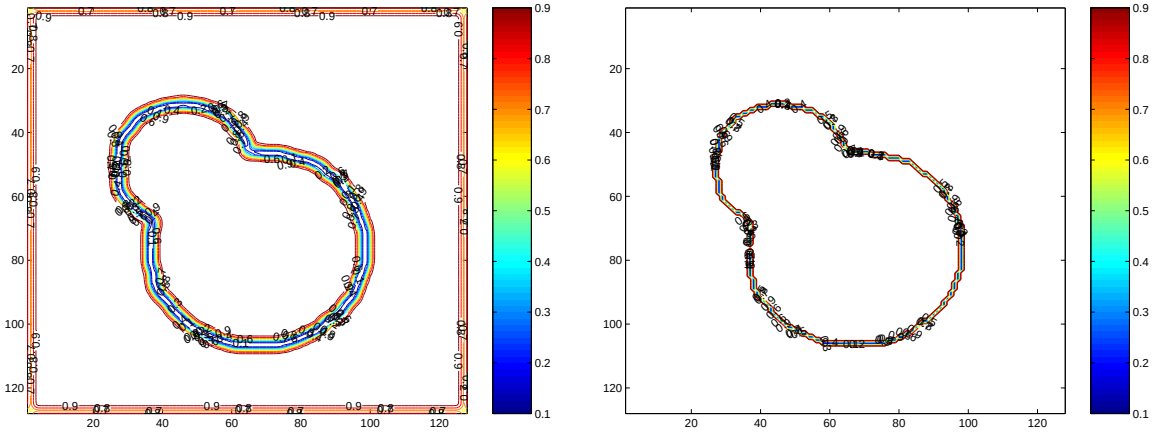


(a) (b) (c) (d) (e) (f)

Figure 3.19: Using Canny edge detector, a) initial, and after b) 1000, c) 2000, d) 3000, e) 4000 and f) 5000 iterations.

certain direction and also speeds it up. The gradient-based detector is used and the image is preprocessed with a 5x5 Gaussian filter, $\sigma \approx 1$.

A spinal (lumbar vertebrae) MRI is tested. The image is difficult with MRI artifacts and noise from neighboring tissues and fat. Furthermore, the boundary of the spinal cord is not always clear and some are broken. To avoid getting stuck at noise or artifacts, a strong advection force is needed. A strong curvature term, on the other hand, is needed to help prevent leakages at broken edges and help the curve to stay smooth. Some results are shown in fig. 3.21. It can be seen that tissue artifacts often still slow down or prevent the front from moving, even with a strong advection term. Meanwhile leakages can be prevented from a stronger curvature term.



(a) (b)

Figure 3.20: a) Contour graph of the edge map using Sobel edge detector, b) using Canny edge detector

However, a strong curvature term also keeps the front from advancing.

As can be seen from this demonstration, there is a trade-off between the advection term and the curvature term. With a strong advection, the front avoids getting stuck at noise, but leakages are more likely. With a strong curvature term, leakages are prevented, but the front potentially misses correct identification of object of interest and also slows down the front propagation.

3.9 Recent Work

While geometric active contour is an improvement over snakes for boundary detection, with the evolving contour's ability to naturally split and merge, problems with noise and ill-defined boundary remain problematic as previously discussed. Recent work aims to overcome these shortcomings through the use of region-based energy functionals. Region-based, as opposed to the classical edge-based energy functionals offer a more global view of the image, making it less

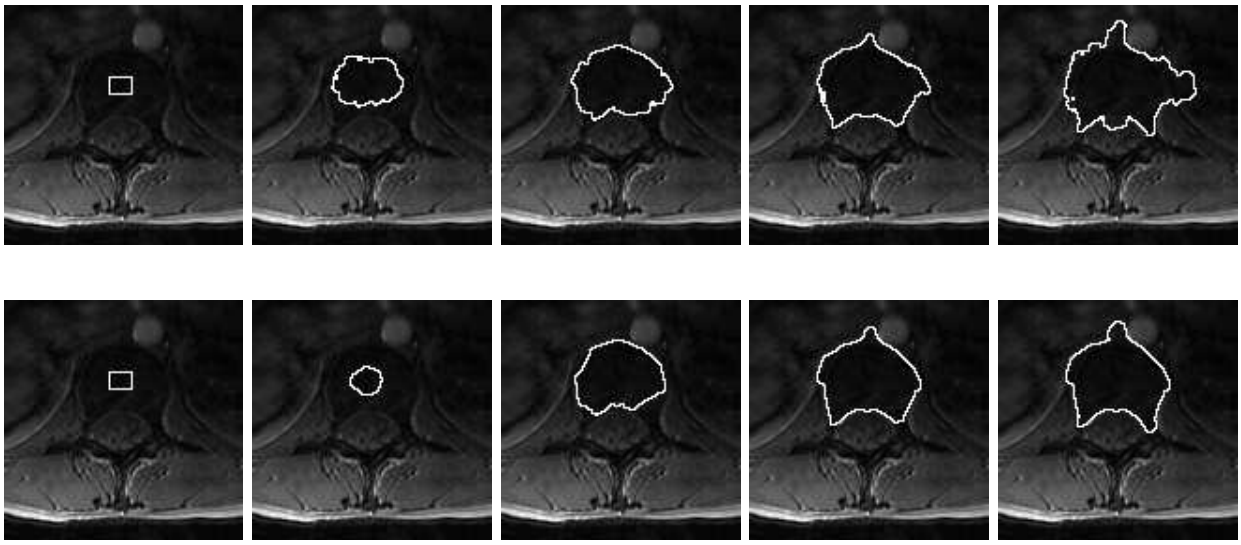


Figure 3.21: Lumbar vertebrae MRI (courtesy of Robarts Research Institute), first row: curve leaked out ($c = -1, \beta = 1, \epsilon = 1$); second row: less leakage ($c = -1, \beta = 3, \epsilon = 1$)

susceptible to the influence of noise and false edges.

Siddiqi et al. [22] proposed the addition of a weighted area functional in Caselles et al.'s geodesic active contour equation [4]. The proposed equation is as follows:

$$\frac{\partial \phi}{\partial t} = \alpha\Big(g\kappa|\nabla\phi| + \nabla g \cdot \nabla\phi\Big) + \frac{1}{2} div\left(\begin{pmatrix} x \\ y \end{pmatrix} g\right)|\nabla\phi| \tag{3.4}$$

The first term and second term are the same as the original geodesic active contour equation, but the original constant advection term is replaced by the area functional. Although this approach improves in dealing with weak edges, there are still cases where "leakages" occur as the curve evolves beyond the weak edge.

In the work by Chan et al. [5], instead of relying on the gradient of the image to detect objects and boundaries, the Mumford-Shah functional is incorporated into the curve evolution equation. The method computes the intensity variance in the selected region and minimizes it to find the

correct segmentation. Although effective, the method applies only to bi-modal images, assuming two homogeneous regions.

Other recent methods incorporate prior processing or information. Xie et al. [27] proposed a region-based method which uses the idea of gradient vector flow [28], but instead of edge map from the image, it incorporates forces obtained through the diffusion of the region segmentation map. Although overcoming noise and weak edges, the method relies on region maps obtained from prior segmentation and depends on the quality of the regions produced.

In [18], Paragios et al. introduce a supervised image segmentation technique which uses given object samples for training to obtain the probability density distribution of the object. Then the probability density distribution is used in computing a region likelihood term to be incorporated into the curve evolution. This way, in addition to having the boundary term, a region-based term is included as well in the curve evolution.

3.10 Summary

Results from an implementation of the geodesic active contour [4] are shown and analyzed in this chapter. It can be seen that although adept at changing topology during evolution, geometric active contours' performance are limited by noise and quality of edges in the image. The initial curve is also constrained to be placed either inside or outside of the object of interest. Preprocessing such as Gaussian smoothing helps with smoother front propagation and a more sophisticated edge detector such as the Sobel operator may improve the results, although care may be required to choose the proper threshold while using the Canny operator. The method is also computationally intensive.

Adjusting the parameters in the equation may help improve the performance of the algorithm, but there is a trade-off between the advection term and the curvature term, and a compromise is need to minimize getting stuck at local optimum and edge leakage.

Multi-resolution methods are often use to speed up computation and may help with avoiding

local optimums. The next chapter discusses some common multi-resolution methods and their applications in active contours.

Chapter 4

Multi-resolution Methods

Multi-resolution methods are commonly used in image denoising, compression, and segmentation. Humans perceive objects as meaningful entities only over certain ranges of scale. Therefore the notion of scale is important in the analysis and derivation of information in the image. Problems such as image denoising and compression can make use of multi-resolution methods to extract meaningful structures in the image through examining corresponding structures at different scales, removing unnecessary and irrelevant details. There are different types of multi-scale representations, including quad-tree, pyramids, scale-space representation, and wavelets [11]. In this section, quad-tree, pyramids, and scale-space and their applications in active contours are discussed.

4.1 Quad-tree

Quad-tree is a tree-like representation of image data where the image is recursively divided into smaller regions [11]. For an image I of size $2^K \times 2^K$, consider a measure f(I) of the grey-level variation (e.g. standard deviation) in any region. Let $I^{(K)}$ denote the image at resolution level K. If $f(I^{(K)})$ is greater than some pre-specified threshold α, then divide $I^{(K)}$ into sub-regions $I_j^{(K-1)}$

(j=1,..,p) according to some rule. The process is done recursively to all sub-images. Fig. 4.1 shows an example of a quad tree, using the difference between the maximum and the minimum intensity as the criteria and with a threshold of 0.4.

(a) (b)

Figure 4.1: Quad-tree demonstration with a difference threshold of 0.4 using MATLAB®

4.2 Pyramid

Pyramid representations of an image give a hierarchy of the image at different scales through a sub-sampling operation with smoothing. For example for an image $2^K \times 2^K$, to construct a pyramid, first there's the sub-sampling, then for each pixel, a smoothing operation (a filter of odd dimension in this example) is performed:

$$f^{(K-1)}(x) = \sum_{n=-N}^{N} c(n) f^{(K)}(2x - n)$$

where c denotes a set of filter coefficients. The choices of c can vary from averaging to Gaussian of different variances.

Because of the use of sub-sampling, dealing with coarser levels of the image means operating on a smaller image and hence less computation. Therefore, pyramids are often used to help reduce computational complexity in algorithms.

Fig. 4.2 shows an example of the pyramid representation.

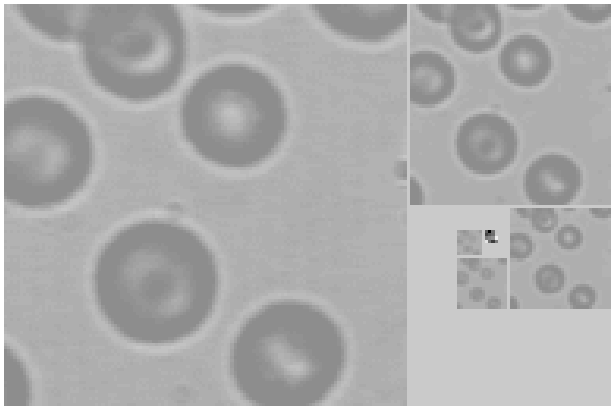


Figure 4.2: Demonstration of the pyramid representation using MATLAB®

4.3 Scale-space Representation

First introduced by Witkin [26], the scale-space representation of an image is an embedding of the original signal into a one-parameter family of Gaussian kernels of increasing widths. The idea is that by smoothing with a Gaussian kernel of increasing standard deviation σ, the structure signal at the corresponding scale gets suppressed with a characteristic length less than σ. In other words, at each successive coarser scale, the fine scale information gets suppressed.

For an image $I : \mathbb{R}^N \to \mathbb{R}$, the scale-space representation $L : \mathbb{R}^N \times \mathbb{R}_+ \to \mathbb{R}$ is defined by $L(\cdot; 0) = I$ and

$$L(\cdot; t) = g(\cdot; t) * I$$

where $t \in \mathbb{R}_+$ is the scale parameter, and $g : \mathbb{R}^N \times \mathbb{R}_+ \backslash \{0\} \to \mathbb{R}$ is the Gaussian kernel:

$$g(x; t) = \frac{1}{(2\pi t)^{N/2}} e^{-\Sigma_{i=1}^{N} x_i^2/(2t)} \quad (x \in \mathbb{R}^N, x_i \in \mathbb{R})$$

The square root of the scale parameter, $\sigma = \sqrt{t}$ is the standard deviation of the kernel g.

The multi-scale spatial derivative is then defined by

$$L_{x^n}(\cdot;t) = \partial_{x^n} L(\cdot;t) = g_{x^n}(\cdot;t) * f$$

where g_{x^n} denotes a derivative of some order n.

4.4 Application to Active Contours

Multi-resolution methods have been employed often to help speed up the computation for active contours. Geiger et al. [7], uses the pyramid structure to help speed up the dynamic programming of the deformable contour. Instead of iterative methods for computing the minimum of the evolution equation, Geiger et al. used dynamic programming which is guaranteed to find the global minimum. The approach is capable of a speedup by a factor of 20, though at the expense of losing the guaranteed optimality characteristics.

Akgul et al. [2] later improved on Geiger et al's work through the use of scale-space representation instead of pyramids. Since no sub-sampling has taken place, the method retains the optimality of the dynamic programming of the contour while achieving faster optimization.

Mignotte et al. [14] also noted the problem with the use of pyramids over dynamic programming of deformable contours, and proposed the use of a multi-grid approach to the algorithm. The multi-grid method involves solving the energy function through an appropriate hierarchy of subspaces, which contain constrained configurations describing the expected solution at different scales. By relaxing these constraints over the subspaces, in effect adjusting the scale, the method offers faster computation. Meanwhile, at each scale level, the image adopts the scale-space representation, where no reduction of image data is applied, and thus with no loss of data information.

In [10], Leroy et al. perform the balloon snake model on an image pyramid and parameters are automatically modified so that, at each scale, the maximal length of the curve is proportional to the image size. Again, it is observed that the computation time for convergence is 55% shorter with the multi-resolution method than the standard one. The authors also attempted a multi-resolution parametrically deformable model using Fourier descriptors in which the curve is first described by a single harmonic, then harmonics of higher frequencies are used so that precision increases with the resolution. Results show that the process from few to more harmonics improve the model's stability and allows it to better capture irregular object boundaries.

These works indicate that multi-resolution methods are mainly used to reduce computation. Multi-resolution methods are also mostly applied to parametric active contours. One work that did apply multi-resolution methods to geometric active contours, not parametric, is by Paragios [19], who used the pyramids method to help speed up a proposed region-based geometric active contour, but without much discussion on how the sub-sampling on coarse resolutions may affect the end results. Edge-based geometric active contour remains very sensitive to noise, and prone to getting stuck at local optimums in such cases. It is the interest of this work to further explore the use of the multi-resolution methods in edge-based geometric active contours.

Chapter 5

Multi-resolution Approach

As mentioned in Chapter 3, geometric active contours still have issues with noise and weak edges. In this thesis, the incorporation of multi-resolution approach to the geometric active contour is explored and modified to improve in these areas. The methodology is described in this section.

5.1 General Idea

There are three main concerns in the basic geometric active contour algorithm: handling of noise, weak edge (blurred, broken), and initialization, which affects the direction of curve flow, inward or outward. These problems can be attributed to the edge-based formulation which is highly localized, and is unable to "see" the object over the global perspective.

To acquire a more global perspective, the use of region rather than edge helps better identify objects of interest, as is seen in recent literature as described in section 3.9. Similarly, multi-resolution analysis is also intuitively a good tool for extracting information about the object of interest, examining objects over different ranges of scale.

Fig. 5.1 shows the algorithm for incorporating the multi-resolution method for geometric active contour. As shown in fig. 5.1, different resolutions of the image are used in the algorithm,

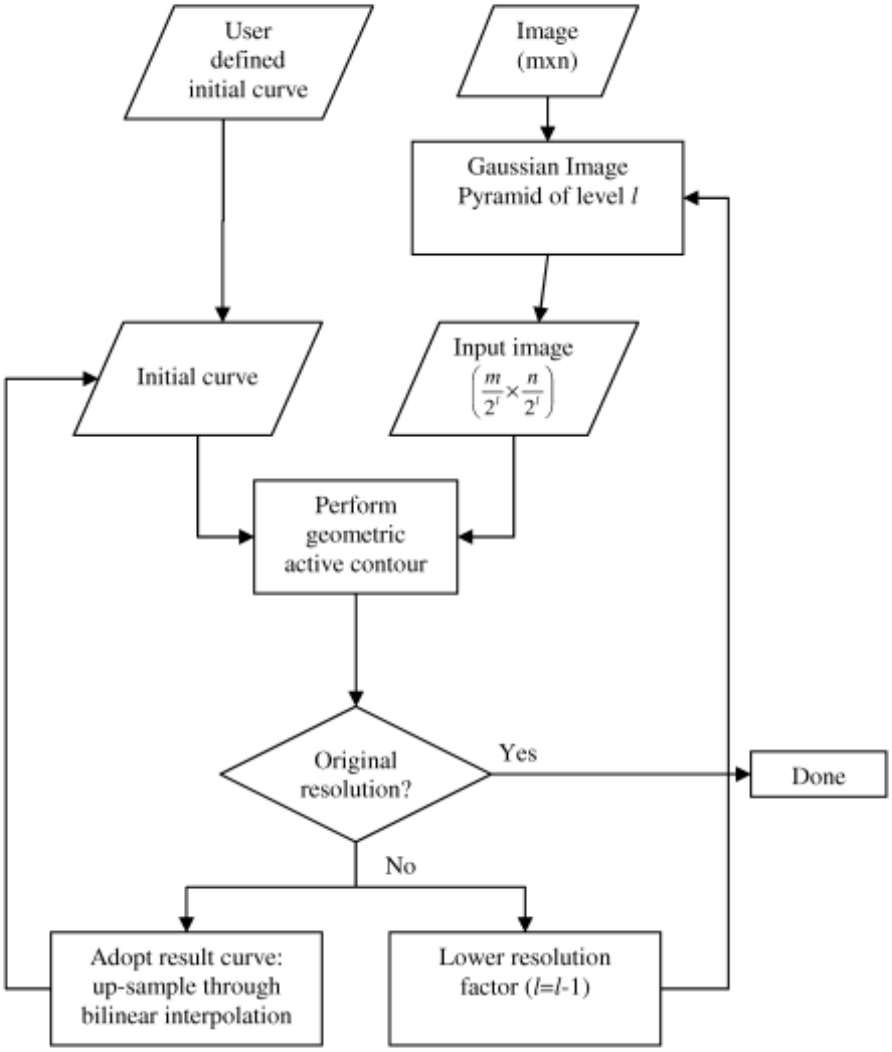


Figure 5.1: Geometric active contour repeatedly applied to different resolution levels

from coarse to original resolution. The results are shown in the next section.

5.2 Results

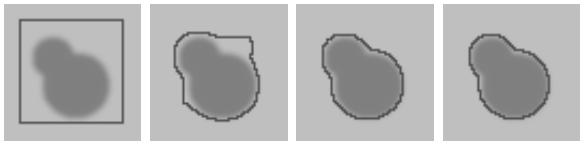
The Gaussian pyramid, as explained in section 4.2, is achieved through sampling the original image and filtering it with a 5x5 Gaussian window (DC gain normalized to 1), σ pre-defined by user, and is then further sampled and smoothed to obtain the different levels. The resizing of the resulting curve from one resolution to a finer one is done through bilinear interpolation.

The algorithm is able to speed up the process considerably while capturing the desired boundary. For example in fig. 5.2, it has significant decrease in computation ($<$ 1 minute), compared with same image without multi-resolution (see fig. 3.8, 3 minutes). Fig. 5.3 shows the noisy synthetic image used before and, when compared with fig. 3.12, it is also much faster (5 minutes vs. 18 minutes). Also, with a large image such as the one in Fig. 5.4, 5.5 (512x512), the computation time is reduced by as much as 18 times.

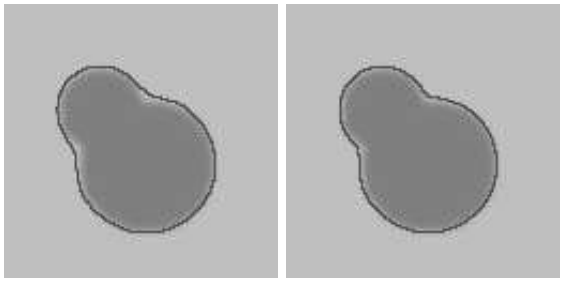
In terms of quality of results, the multiresolution approach shows better results. Note that the degree of smoothing in both multi-resolution and non-multi-resolution approaches are the same at the original resolution, thus ensuring a more compatible comparison. In the case of the noisy image, as seen in Fig. 5.3, multi-resolution also yields a cleaner result, with fewer spots caused by noise. This is due to the successive smoothing done in the multi-resolution method. In the left ventricular angiography image, fig. 5.5, without multi-resolution, has the edge leakages, whereas with multi-resolution, in fig. 5.4, the boundary is successfully captured.

5.3 Discussion

The obvious benefit of the multi-resolution approach is the considerable reduction in computation time, as demonstrated with previous tests. This is due to the computation of fewer pixels when working at coarse resolution levels. Meanwhile, the improvement in convergence is more subtle.



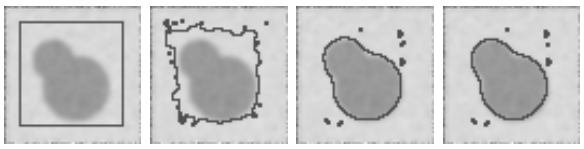
(a)



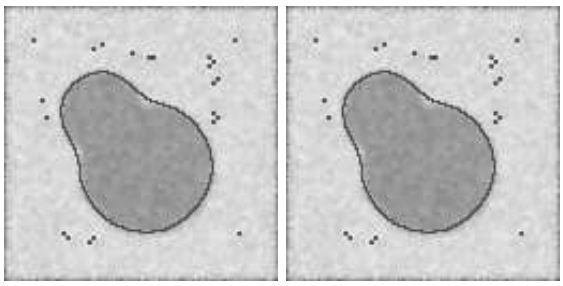
(b)

Figure 5.2: Test image (from Fig. 3.8), $\sigma = 1$, a) at resolution level 1; b) at level 0 (full resolution) (< 1 minutes)

The determination of the appropriate smoothing parameter is vital to the success of the algorithm, and in this respect, it is similar to the non-multi-resolution approach, where too much smoothing can cause significant loss of edge information, causing edge leakage, and too little smoothing may cause front to be stuck at local optimum. However, there is a noticeable improvement in capturing the right boundary in the multi-resolution approach. By starting at a coarse resolution level, local optimums are removed, and the curve approaches the boundary with no interference, and the front, at each successive resolution level, can adjust to edge information at finer levels. It is also interesting to note that the sub-sampling also helps in preventing edge leakage. At coarse resolution levels, weak edges are less noticeable and the curve is more likely to stay smooth and not leak out. See fig. 5.6 for a sub-sampled image with the same amount of smoothing. The algorithm succeeded without edge leakage.



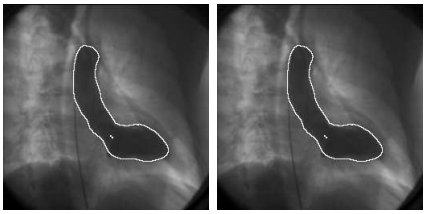
(a)



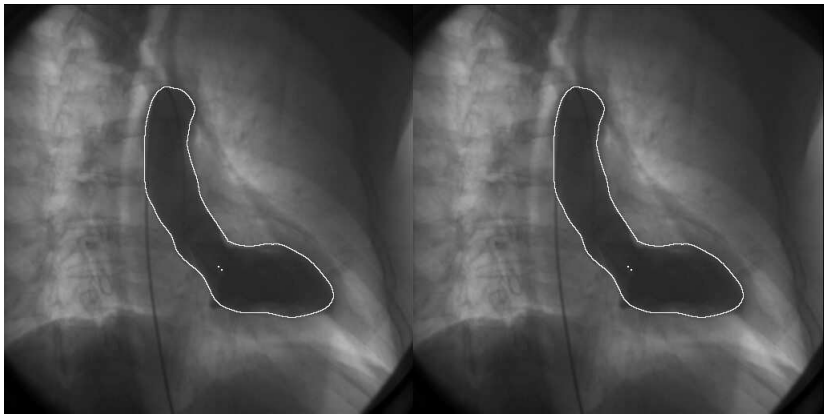
(b)

Figure 5.3: Test image (from Fig. 3.12) with gaussian noise (mean=0.1, s.d.=0.005), $\sigma = 1$, a) at level 1; b) at level 0 (full resolution) (5 minutes)

(a)

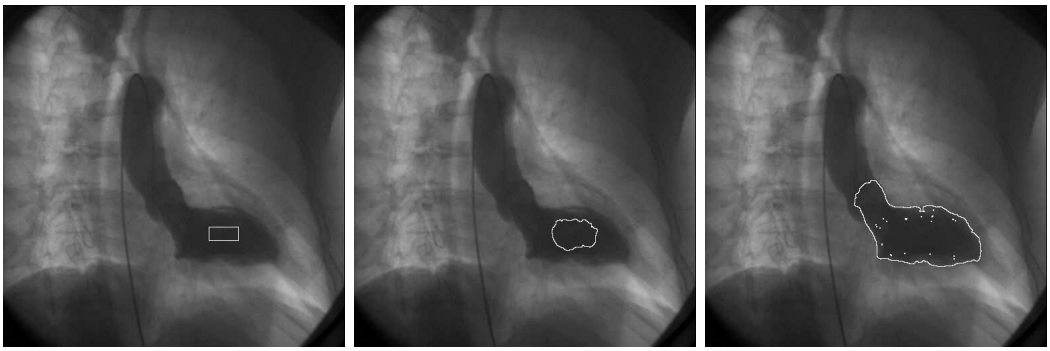


(b)

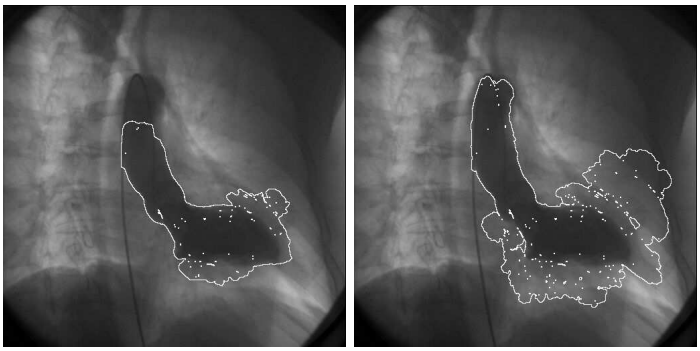


(c)

Figure 5.4: Ventricular angiography (courtesy of Fu Jin (fjin@engmail.uwaterloo.ca)), $\sigma = 0.6$, a) at resolution level 2; b) at level 1; c) at level 0 (full resolution). (10 minutes)



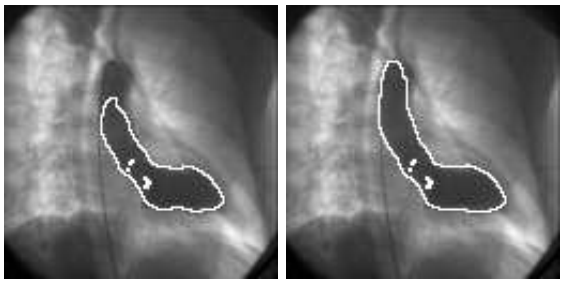
(a) (b) (c)



(d) (e)

Figure 5.5: Left ventricular angiography (from Fig. 5.4) without multi-resolution, $\sigma = 0.6$, a) initial, and after b) 2000, c) 20000, d) 40000, and e) 58000 iterations. (180 minutes)

(a) (b) (c)



(d) (e)

Figure 5.6: Left ventricular angiography (from Fig. 5.4), sub-sampled (128x128), $\sigma = 0.6$, a) initial, and after b) 2000, c) 12000, d) 24000, and e) 36000 iterations. (8 minutes)

Chapter 6

Conclusions

In this work, active contours as a segmentation method are introduced and discussed. Active contours at first are parametric curves that evolve according to image features such as edges, and level sets are later incorporated as an improvement. Active contours that use level sets are known as geometric active contours, and are the focus of this study. It is an attractive method for medical image segmentation as it is able to capture the object of interest in one continuous curve.

As geometric active contours are based in partial differential equations, careful numerical methods are needed to implement the method properly. First order differences are used but depending on the general direction of the curve movement, more sophisticated schemes are needed. Such schemes are discussed in sec. 2.3.1. Re-initialization of the level set function is also needed periodically to ensure the proper evaluation of the PDE, and constraints are placed on the time interval. Since the algorithm requires iterating through the PDE, it is computationally intensive, and the narrow band propagation algorithm is introduced and used as a way to speed up the process.

The robustness of the algorithm is tested through a series of tests, involving both synthetic images and medical images. Common problems occur when edges are blurred and broken, causing

the curve to leak out. Also, noise causes the curve to be stuck before getting close to the boundary. Smoothing can be very helpful and choosing the appropriate smoothing parameter is vital to the success of the algorithm. Parameters in the equation can also be helpful in controlling the behavior of the algorithm. However, a trade-off exists between the advection term and the curvature term, where strong advection avoids getting stuck at local optimums but may lead to edge leakage, and the curvature term helps prevent edge leakage but slows down or even stops front propagation.

The multi-resolution approach, specifically, a Gaussian image pyramid, is incorporated into the geometric active contour. The approach speeds the process up considerably, as much as 18 times, and with successive smoothing and sub-sampling, performance improves. However, similar to the single-resolution approach, choosing the appropriate smoothing parameter is important.

6.1 Contributions

This work has implemented the geometric active contour technique in [4], demonstrated its performance under different settings such as noise, contrast, and parameters to the algorithm. It proposed to incorporate the Gaussian pyramid into the geometric active contour algorithm and found the incorporation to improve both the speed and segmentation results of the algorithm.

Through the implementation of the algorithm, the limitations of the algorithm are thoroughly examined. Also, although a popular research area, most papers on geometric active contours empirically determine the parameters in the equation and do not have selection criteria. This work sets to examine the impact of parameters on the equation. Results show that while the adjustment of advection and curvature terms may help, the localized nature of the edge-based method makes it difficult for the parameters to have an impact, since certain areas of the image would appreciate a larger advection term while others may desire a smaller one to prevent edge leakage. There is a constant trade-off between the two terms.

This work also explored the incorporation of the multi-resolution method of Gaussian pyramids into the algorithm. Although similar multi-resolution methods have been applied to para-

metric active contours, as discussed in section 4.4, there is not much literature on its application to the edge-based geometric active contour. Such an approach was mentioned or suggested in papers such as [4], but not implemented. This work implemented the approach by performing the geometric active contour method at different resolution levels and showed that similar to the multi-resolution methods applied to parametric active contours, the multi-resolution can greatly increase the computation speed without sacrificing performance. In fact, results show that with successive smoothing and sub-sampling, performance often improves.

6.2 Future Work and Recommendations

Although parameters in the geometric equation can help shape the behaviour of the algorithm, a trade-off exists between the advection and the curvature term. Even with the multi-resolution approach, the smoothing is still global and although better at avoiding getting stuck at local optimums, edge leakage is possible with weak edges. More emphasis should be placed on using adaptive edge detectors and smoothing operators on the image to avoid noise and weak edge problems. Also, region-based approaches is promising in future work on active contour segmentation methods.

References

[1] D. Adalsteinsson and J. A. Sethian. A fast level set method for propagating interfaces. *Journal of Computational Physics*, 118(2), 1995.

[2] Y. S. Akgul and C. Kambhamettu. A scale-space based approach for deformable contour optimization. In *Second International Conference on Scale-Space Theories in Computer Vision*, pages 410–422, Kerkyra, Greece, September 1999.

[3] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6), 1986.

[4] V. Caselles, R. Kimmel, and Guillermo Sapiro. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–79, 1997.

[5] T. Chan and L. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, February 2001.

[6] D. Chop. Computing minimal surfaces via level set curvature flow. *Journal of Computational Physics*, 106(1):77–91, 1993.

[7] D. Geiger, A. Gupta, L.A. Costa, and J. Vlontzos. Dynamic programming for detecting, tracking, and matching deformable contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(3):294–302, 1995.

[8] R. Gonzalez and R. Woods. *Digital Image Processing.* Addison Wesley, 1992.

[9] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–31, 1988.

[10] B. Leroy, I. Herlin, and L. Cohen. Multi-resolution algorithms for active contour models. In *Proceedings 12th International Conference on Analysis and Optimization of Systems: Images, Wavelets and PDE's*, Paris, France, June 1996.

[11] T. Lindeberg. Scale-space theory: A basic tool for analysing structures at different scales. *Journal of Applied Statistics*, 21(2):224–270, 1994.

[12] C. Lundström. Segmentation of medical image volumes. Master's thesis, Linköping University, SE-581 83 Linköping, Sweden, November 1997. LiTH-ISY-EX-1864.

[13] R. Malladi, J. A. Sethian, and B. C. Vemuri. Shape modeling with front propagation: A level set approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):158–175, 1995.

[14] M. Mignotte and J. Meunier. An unsupervised multiscale approach for the dynamic contour-based boundary detection issue in ultrasound imagery. In *International Conference on Computer Vision, Pattern Recognition and Image Processing*, volume 2, pages 366–369, Atlantic City, New Jersey, USA, March 2000.

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

[16] S.J. Osher and J.A. Sethian. Fronts propagation with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79:12– 49, 1988.

[17] R. L. Panton. *Incompressible Flow.* John Wiley & Sons, Inc., 1984.

[18] N. Paragios and R. Deriche. Geodesic active regions and level set methods for supervised texture segmentation. *International Journal of Computer Vision*, 46(3), 2002.

[19] N. K. Paragios. *Geodesic Active Region and Level set Methods: Contributions and Applications in Artificial Vision*. PhD thesis, University of Nice/Sophie Antipolis, INRIA, January 2000.

[20] R. A. Robb. *Biomedical Imaging, Visualization, and Analysis*. Wiley-Liss, Inc., 2000.

[21] J. Sethian. A fast marching level set method for monotonically advancing fronts, 1996.

[22] K. Siddiqi, Y. Lauziere, A. Tannenbaum, , and S. Zucker. Area and length-minimizing flows for shape segmentation. *IEEE Transactions on Image Processing*, 7(3):433–443, 1998.

[23] P. Suetens. *Fundamentals of Medical Imaging*. Cambridge University Press, 2002.

[24] A. Tsai, A. J. Yezzi, W. M. Wells, C. Tempany, D. Tucker, A. Fan, W. E. Grimson, and A. S. Willsky. A shape-based approach to the segmentation of medical imagery using level sets. *IEEE Transactions on Medical Imaging*, 22(2):137–154, 2003.

[25] J. J. West. Application of the level set method to hydrocephalus: Simulating the motion of the ventricles. Master's thesis, University of Waterloo, Waterloo, Ontario, Canada, 2004.

[26] A. Witkin. Scale-space filtering: A new approach to multi-scale description. In *IEEE International Conference on Accoustics, Speech, and Signal Processing*, volume 9, pages 150–153, March 1984.

[27] X. Xie and M. Mirmehdi. Rags: Region-aided geometric snake. *IEEE Transactions on Image Processing*, 13(5):640–652, 2004.

[28] C. Xu and J. L. Prince. Gradient vector flow: A new external force for snakes. In *IEEE Proceedings Conference on Computer Vision Pattern Recognition*, pages 66–71, June 1997.

[29] Y. Zhang, M. Brady, and S. Smith. Segmentation of brain MR images through a hidden Markov random field model and the expectation-maximization algorithm. *IEEE Transaction on Medical Imaging*, 20(1):45–57, 2001.