Decoupled Deformable Model For 2D/3D Boundary Identification

by

Akshaya Kumar Mishra

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2010

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.

Abstract

The accurate detection of static object boundaries such as contours or surfaces and dynamic tunnels of moving objects via deformable models is an ongoing research topic in computer vision. Most deformable models attempt to converge towards a desired solution by minimizing the sum of internal (prior) and external (measurement) energy terms. Such an approach is elegant, but frequently mis-converges in the presence of noise or complex boundaries and typically requires careful semi-dependent parameter tuning and initialization. Furthermore, current deformable model based approaches are computationally demanding which precludes real-time use.

To address these limitations, a decoupled deformable model (DDM) is developed which optimizes the two energy terms separately. Essentially, the DDM consists of a measurement update step, employing a Hidden Markov Model (HMM) and Maximum Likelihood (ML) estimator, followed by a separate prior step, which modifies the updated deformable model based on the relative strengths of the measurement uncertainty and the non-stationary prior. The non-stationary prior is generated by using a curvature guided importance sampling method to capture high curvature regions. By separating the measurement and prior steps, the algorithm is less likely to mis-converge; furthermore, the use of a non-iterative ML estimator allows the method to converge more rapidly than energy-based iterative solvers.

The full functionality of the DDM is developed in three phases. First, a DDM in 2D called the decoupled active contour (DAC) is developed to accurately identify the boundary of a 2D object in the presence of noise and background clutter. To carry out this task, the DAC employs the Viterbi algorithm as a truncated ML estimator, curvature guided importance sampling as a non-stationary prior generator, and a linear Bayesian estimator to fuse the non-stationary prior with the measurements. Experimental results clearly demonstrate that the DAC is robust to noise, can capture regions of very high curvature, and exhibits limited dependence on contour initialization or parameter settings. Compared to three other published methods and across many images, the DAC is found to be faster and to offer consistently accurate boundary identification.

Second, a fast decoupled active contour (FDAC) is proposed to accelerate the convergence rate and the scalability of the DAC without sacrificing the accuracy by employing computationally efficient and scalable techniques to solve the three primary steps of DAC.

The computational advantage of the FDAC is demonstrated both experimentally and analytically compared to three computationally efficient methods using illustrative examples.

Finally, an extension of the FDAC from 2D to 3D called a decoupled active surface (DAS) is developed to precisely identify the surface of a volumetric 3D image and the tunnel of a moving 2D object. To achieve the objectives of the DAS, the concepts of the FDAC are extended to 3D by using a specialized 3D deformable model representation scheme and a computationally and storage efficient estimation scheme. The performance of the DAS is demonstrated using several natural and synthetic volumetric images and a sequence of moving objects.

Acknowledgements

I have heard stories about the relationship between a student and a supervisor, and have only now realized how important such a relationship is for the professional growth of a student. I sincerely believe that this thesis would not be possible without the active and loving support of my supervisors, Prof. David A. Clausi and Prof. Paul W. Fieguth. Both have played equally important and complementary roles in my professional development, providing all kinds of support for me, be it technical, personal or financial. Furthermore, I would like to thank them for giving me complete freedom in research while still guiding me towards a fruitful path to completing my degree. Finally, I would particularly like to give special acknowledgement to my supervisors for continuously helping me to improving my English and never showing disheartenment to my initially poor writing skills.

I would like to thank my committee members Prof. Edward Vrscay, Prof. John Zelek and Prof. Jeff Orchard for their agreeing to serve as my committee members and providing valuable comments on my comprehensive proposal. I would like to thank the external examiner, Prof. Scott Acton for providing valuable suggestions. Also, I would like to thank Prof. Thomas O. Binford for motivating me to go for higher study.

Another important part of my professional growth is my colleagues in the Vision and Image Processing research group. I would like to thank them for creating a cordial study atmosphere. Particularly, I would like to thank Alex for discussing about a variety of interesting topics and helping me improve my technical writing skills.

I would like to thank the Graduate Studies Office, Faculty of Engineering, and Department of Systems Design Engineering, University of Waterloo for providing financial support.

Finally and most importantly, I would like to thank my wife, Sini. Her patience, love and understanding were instrumental in the successful completion of my thesis.

Dedication

This work is dedicated to the
most important people in my life
Sini and my parents.

Contents

List of Tables

List of Figures

List of Algorithms

Nomenclature

Syntax	Definition
1D	one dimension
2D	two dimensions
3D	three dimensions
P	probability
i, j, and k	indices
$z_j \in [\frac{-u}{2} : \frac{u}{2}]$	states of a Hidden Markov Model
u	number of nodes along a normal
q	number of nodes required to represent $\mathbf{v}$
s_j	normalized arclength
$\alpha, \beta, \alpha_i, \beta_i$	constants
ϵ	a small quantity
$\mathcal{E}_{tot}$	total energy
$\mathcal{E}_{int}$	internal energy
$\mathcal{E}_{ext}$	external energy
$\mathcal{F}_{int}$	internal force
$\mathcal{F}_{ext}$	external force
$\mathcal{F}_{balloon}$	balloon force
$\mathcal{F}_{GVF}$	gradient vector flow force
$\mathcal{F}_{VFC}$	vector filed force
$\mathbf{v}$	random vector
v_j	j^{th} element of vector $\mathbf{v}$
$v(s_j)$	the value of v at s_j

Syntax	Definition
$\mathbf{v}^m$	measured boundary
$\mathbf{v}^e$	estimated boundary
$\mathbf{v}^c$	converged boundary
$\boldsymbol{\kappa}$	curvature
$\boldsymbol{n}$	normal
Δ	differential operator
Δv_{max}	maximum length of a segment of the deformable model $\mathbf{v}$
Δv_{min}	minimum length of a segment of the deformable model $\mathbf{v}$
Δv_j	length of a discrete segment
$\Omega(j)$	set of neighbors of the node j
t_k	k^{th} traingle
$\mathcal{TV}(t_k)$	the vertices of the triangle t_k
$\mathcal{VT}(v_j)$	all triangles incident to the vertex v_j
Λ	prior
Q	constraint matrix
R	measurement noise covariance matrix
$\mathbf{r}$	diagonal elements of R

Glossary of Terms

Abbreviation	Definition
ACWE	active contour without edge
ASD	average shortest distance
DAC	decoupled active contour
DAS	decoupled active surface
DDM	decoupled deformable model
FDAC	fast decoupled active contour
GVF	gradient vector flow
HMM	hidden Markov model
NPAC	non-parametric active contour
PAC	parametric active contour
PIG	Poisson inverse gradient
TS	traditional snake
VFC	vector field convolution

Chapter 1

Introduction

Automatically identifying the exact boundary, known as a contour in 2D and a surface in 3D, of a 2D/3D object or the tunnel of a moving 2D object from a three-dimensional image, in the presence of noise and background clutter, has many applications in biomedical image analysis [1, 2], visual tracking [3, 4], content based image and video retrieval [5, 6], robotics [7], image and video composition [8, 9], visualization [10, 11], surface reconstruction [12], deformable template matching [13, 14] and morphing [15–17]. For example, as illustrated in Figs. 1.1, 1.2, 1.3 and 1.4, boundary extraction techniques are essential to segment and quantify important structures of 2D/3D images for successful recognition or diagnosis.

1.1 Overview of Challenges

Despite the fact that the boundary of an object in a 2D/3D image can be easily detected using several low level edge detection techniques (such as Canny and Laplacian edge detectors), precise continuous boundary identification needs specific high level knowledge. Such a model based identification task can be achieved by using the concepts of deformable model [20].

Developing a generic method to perform such boundary identification tasks is a challenging problem for three main reasons.

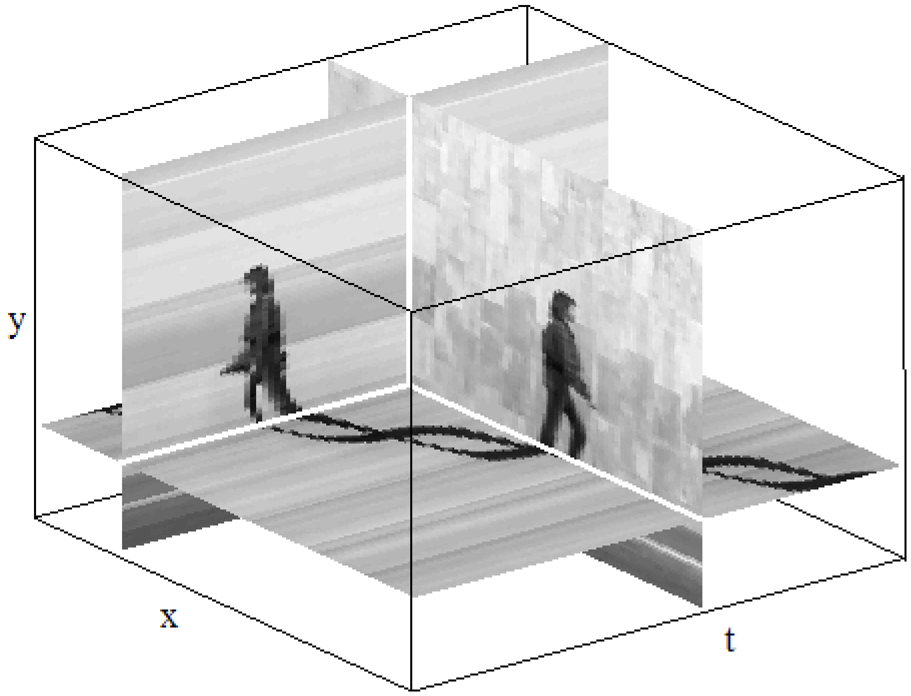


Fig. 1.1: The walking image sequences of a man: identifying the tunnel of the walking man has several applications in computer vision [18].

1. The observed 2D/3D images are acquired using different imaging devices under various imaging environments. These imaging techniques include fluoroscopy, projectional radiographs, magnetic resonance (MR), positron emission tomography (PET), computer tomography (CT) and ultrasound. The observed 2D/3D images collected from different sources contain noise and background clutter. From an optimization point-of-view, the noise and background clutter act as potential local minima and existing methods are sensitive to such local minima.

2. Real-world objects have high curvature contours and surfaces. The state-of-the-art boundary extraction methods are generally not able to identify high curvature regions of a body.

3. Computational and storage burden hinder the practical application of existing boundary extraction techniques.

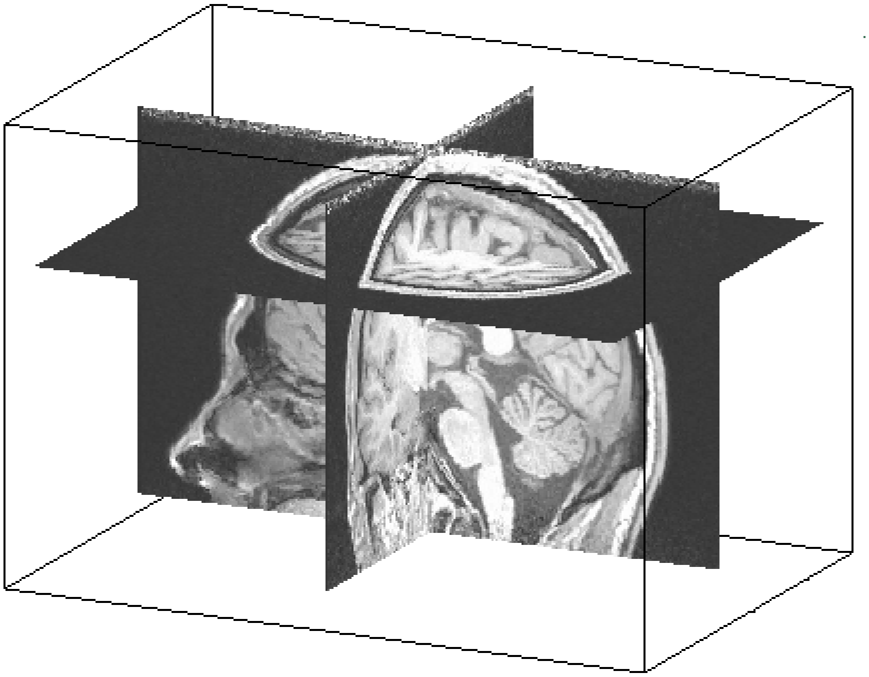


Fig. 1.2: MRI of a human brain: producing a 3D segmentation of such data has countless applications in diagnosis and treatment.

Current boundary extraction methods and their limitations are described next.

1.2 Current Methods

Deformable model based boundary identification techniques, known as active contours or snakes in 2D and active surfaces in 3D, have become popular after the seminal work of Kass et al. [20]. The principal idea in deformable model based boundary extraction methods is to minimize the sum of internal (prior) and external (image-based) energies to obtain an optimum boundary [20–26]. The internal energy typically asserts a first- or second-order or both first- and second-order smoothness constraints on the boundary, whereas the external energy applies an attractive "force" on the boundary, typically towards areas of high image gradient. Since the original development of active contour or surface based methods [20–23], many variations have been developed, falling broadly into two classes:

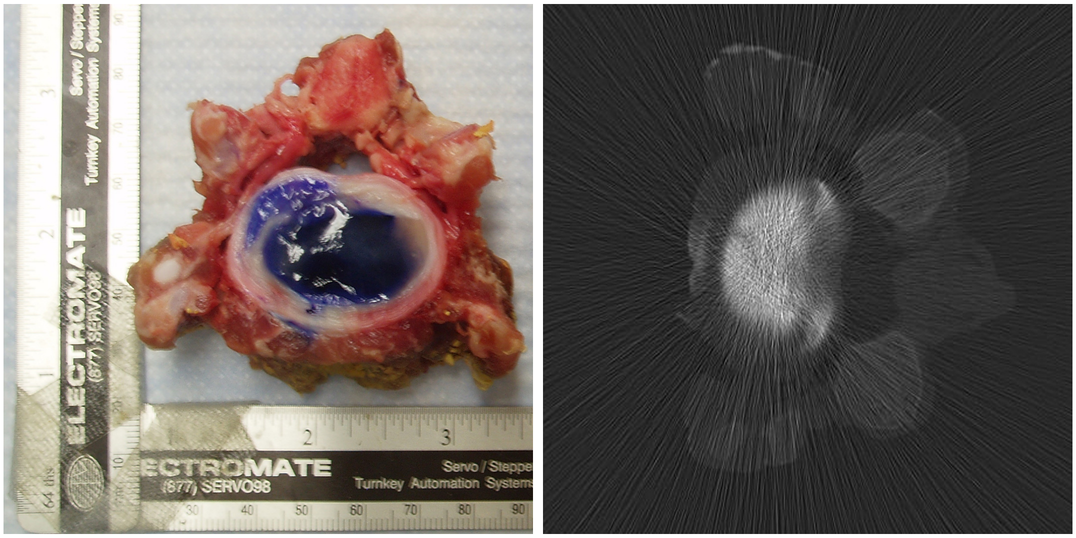


Fig. 1.3: A photo of a porcine functional spinal unit with the contrast agent (shown in blue) injected into the intervertebral disc (left), peripheral CT image (right) of a slice. These images are used to study the deformation of the functional spinal unit as a function of compressive loads [19].

i) parametric deformable models [20–22, 27–34], and ii) non-parametric or geometric deformable models [35–47]. Parametric deformable models are usually modeled using an explicit parametric model describing a contour or surface of a single 2D or 3D object. Geometric active contours are implicitly modeled as a zero level set function in a higher dimension and such methods are suitable for identifying boundaries of multiple objects.

Despite the large number of existing approaches, the parametric deformable models are generally ineffective to handle the problems associated with image noise contamination, complex high curvature boundaries, algorithmic parameter sensitivity, initialization sensitivity, ineffective stopping criteria and slow convergence rate. Non-parametric approaches are initialization independent and are able to handle high curvature regions and topology naturally. However, non-parametric active contours are comparatively slower and more sensitive to noise compared to parametric methods. Several faster techniques, including sparse field [48], fast marching [49] and multi-scale [50, 51] techniques are available to accelerate the computational performance of traditional geometric active contours. However, the computational complexity of geometric active contours are proportional to the total number of image pixels, while the computational complexity of parametric active contours

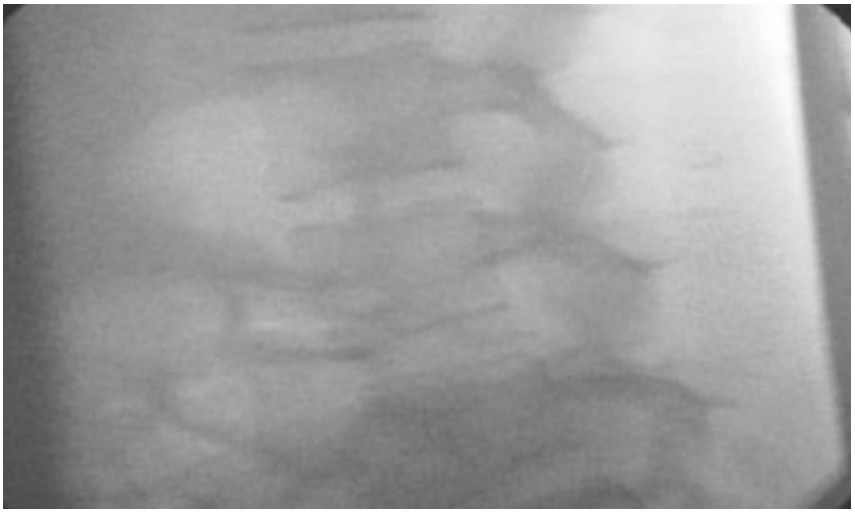


Fig. 1.4: A slice of a video fluoroscopic image of a lumbar spine. Segmenting and tracking the spine help to study spine dynamics.

are related to number of active contour vertices. The number of pixels in an image is much larger than the number of active contour vertices. Hence, non-parametric methods use far more computational resources compared to parametric methods. Whereas non-parametric approaches segment a single object into any number of sub-objects since the deformable boundary is not parameterized using a single deformable model, parametric deformable models are able to find the single continuous boundary of an object from its sparse measurement. As such, parametric deformable models are useful to find the boundary of a single object. In contrast, the desired boundaries cannot always be identified by employing thresholding, region growing or non-parametric active contour approaches, as these techniques often produce spurious edges and gaps, rather than a continuous curve, due to their local and non-parametric nature.

Parametric deformable models, which use a global parametric model [20, 52, 53], are a highly effective class in the computer vision literature for boundary extraction problems. Several modifications to the original parametric deformable idea [20] have been developed [21–24, 24, 28, 54, 55]. Most [21, 22, 28, 54] of these focus on modifying the external energy to increase the capture range of the deformable model such that a deformable model initialized far from the true boundary can be attracted towards the true boundary. However, the non-convergence and computational issues of the original deformable model have been generally ignored.

Current deformable model based boundary extraction techniques are iterative in nature and the iterative solution techniques are prone to being trapped at a local minimum and are slow to converge. The outliers act as potential local minima and the real-world data collected using various imaging techniques are not free of outliers. Therefore, the local minimum trap is considered as the common reason of the failure of the present deformable model based methods.

1.3 The Proposed Approach

In tackling these issues, this thesis introduces a novel parametric decoupled deformable model (DDM), that shares the same formulation and origin of conventional deformable models. However, instead of discretizing and iteratively minimizing the total energy, as in most approaches, the proposed method alternately minimizes the external energy within a specified region, then separately asserts the prior constraints to force the boundary to satisfy required smoothness. The adaptivity of the technique to sharp corners is satisfied by importance sampling [56] on the basis of curvature. Essentially, DDM finds the boundary of a 2D/3D object within a specified solution space using following three steps.

- Step 1: identifies an approximate boundary, termed as the measured boundary using an external energy and an inter-point constraint.
- Step 2: generates a non-stationary prior based on the curvature of the measured boundary to capture high curvature regions.
- Step 3: finds an estimated boundary by asserting the non-stationary prior on the measured boundary.

DDM is less sensitive to parameter tuning as the parameters are derived implicitly from the image curvature and gradient.

1.4 Thesis Outline

The remainder of the thesis is organized as follows. Chapter 2 reviews in more detail existing deformable model based boundary extraction techniques and comments on the merits and drawbacks of each method. Chapter 3 formulates the boundary extraction problem as a maximum-a-priori (MAP) problem using a Bayesian framework and develops a concept, termed as decoupled deformable model (DDM), to efficiently solve the MAP problem for identifying the boundary of an object.

Chapter 4 proposes a decoupled active contour (DAC), a DDM in 2D to precisely identify the boundary of a complex 2D object in the presence of noise and background clutter. The performance of the DAC compared to one non-parametric and two parametric boundary extraction techniques are clearly demonstrated in this chapter. Chapter 5 proposes a faster approach, named as fast decoupled active contour (FDAC) while maintaining the DAC's segmentation accuracy and improving computational performance. As such the FDAC is scalable to higher dimensions.

Chapter 6 develops a decoupled active surface (DAS), an extension of FDAC from 2D to 3D, for identifying the outer surface of a static 3D object or the tunnel of a moving 2D object. Finally, the summary and future direction of DDM is provided in Chapter 7.1.

Chapter 2

Background Theory

This chapter provides a review of necessary background literature related to the work developed in this dissertation. Specifically, this chapter discusses the theory and limitations of present deformable model based boundary (contour/surface) extraction methods. Since the proposed research stems from the fundamental concepts behind parametric deformable models, a detailed discussion of these models is provided. First, the mathematical formulation of the parametric deformable model energy functional to identify the object boundary is derived. Second, a brief overview of non-parametric deformable models is presented. Third, the limitations of present deformable model based boundary extraction techniques are discussed. Finally, the objectives of this thesis are outlined.

2.1 Deformable Model For Boundary Identification

Generally, the boundary is defined as a continuous border that separates two dissimilar regions. The boundary of an object is often identified based upon either some region similarity [57–59] or discontinuity criteria, however there is substantial overlap between these two criteria [60, 61]. Among these two segmentation categories, the region discontinuity (also know as deformable model) based segmentation methods [17, 20, 21, 34, 37, 43, 62–70] have been gaining popularity due to their ability to represent an object using a physical model.

Deformable models, popularly known as active contours in 2D and active surfaces in

3D, are energy minimizing splines [20]. The deformable models evolve in a constrained environment to achieve a certain goal by minimizing an energy functional $\mathcal{E}_{tot}$ [20]. When the goal is to identify the boundary of a 2D/3D object, the total energy of the deformable model is expressed as a function of the observed boundary and some prior assumption about the true boundary:

$$\mathcal{E}_{tot} = \Im(\mathcal{E}_{int}, \mathcal{E}_{ext}). \tag{2.1}$$

The external energy $\mathcal{E}_{ext}$ (also known as attractive energy or stopping potential) represents the measurement and pulls the deformable model towards the object boundaries. On the other hand, the internal energy $\mathcal{E}_{int}$ (also know as shrinkage energy) attracts the deformable model towards the prior.

As DDM is developed based upon the theory of deformable models and the performance of DDM is evaluated against four state-of-the-art parametric deformable models and one state-of-the-art non-parametric model, therefore a comprehensive review of the deformable models is presented next.

2.2 Parametric Deformable Model

2.2.1 Mathematical Model Representation

Parametric deformable models are free form (there are no global constraints, except local smoothness constraints) active models which can adapt themselves into various shapes. As shown in Fig. 2.1, a parametric deformable model [20–22] in 2D is represented using a continuous 2D parametric curve

$$\mathbf{v} = \{v(s)\} = [x(s), y(s)], s \in [0, 1] \tag{2.2}$$

or a discrete parametric curve

$$\mathbf{v} = \{v_j\} = \{v(s_j)\} = [x_j, y_j] = [x(s_j), y(s_j)], \quad j \in [1, q], s_j \in [0, 1]. \tag{2.3}$$

Similarly, as shown in Fig. 2.2, a parametric deformable model [20–22] in 3D is represented using a continuous parametric surface

$$\mathbf{v} = \{v(a, b)\} = [x(a, b), y(a, b), z(a, b)], a \in [0, 1], b \in [0, 1], \tag{2.4}$$

or a discrete parametric surface

$$\mathbf{v} = \{v_{i,j}\} = [x_{i,j}, y_{i,j}, z_{i,j}], i \in [1, q], j \in [1, q], \tag{2.5}$$

where the normalized arclength s parameterizes the continuous deformable model in 2D and the parameters $[a, b]$ parameterize the continuous deformable model in 3D. In discrete space, the q vertices (v_j) are used to represent the deformable model in 2D, and a $q \times q$ grid is used to represent the vertices $(v_{i,j})$ of the deformable model in 3D.

The parametric deformable model evolves in space or time to minimize the total energy $\mathcal{E}_{tot}$ of the parametric deformable model $\mathbf{v}$. The total energy is expressed as a sum of an internal energy $\mathcal{E}_{int}$ and an external energy $\mathcal{E}_{ext}$ and mathematically written as:

$$\mathcal{E}_{tot}(\mathbf{v}) = \mathcal{E}_{int}(\mathbf{v}) + \mathcal{E}_{ext}(\mathbf{v}). \tag{2.6}$$

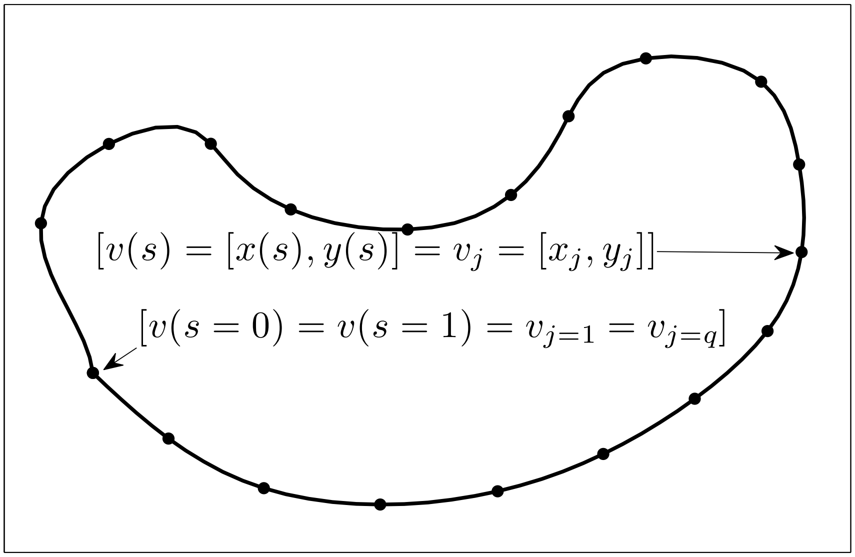

$[v(s) = [x(s), y(s)] = v_j = [x_j, y_j]]$
$[v(s=0) = v(s=1) = v_{j=1} = v_{j=q}]$

Fig. 2.1: A parametric active contour is represented using a random field $\mathbf{v} = \{v(s)\}, s \in [0, 1]$ in the continuous domain or $\mathbf{v} = \{v_j\}, j \in [1, q]$ in the discrete domain. The term s is the normalized arclength, j is an index, and x, y are the components of the random vector v.

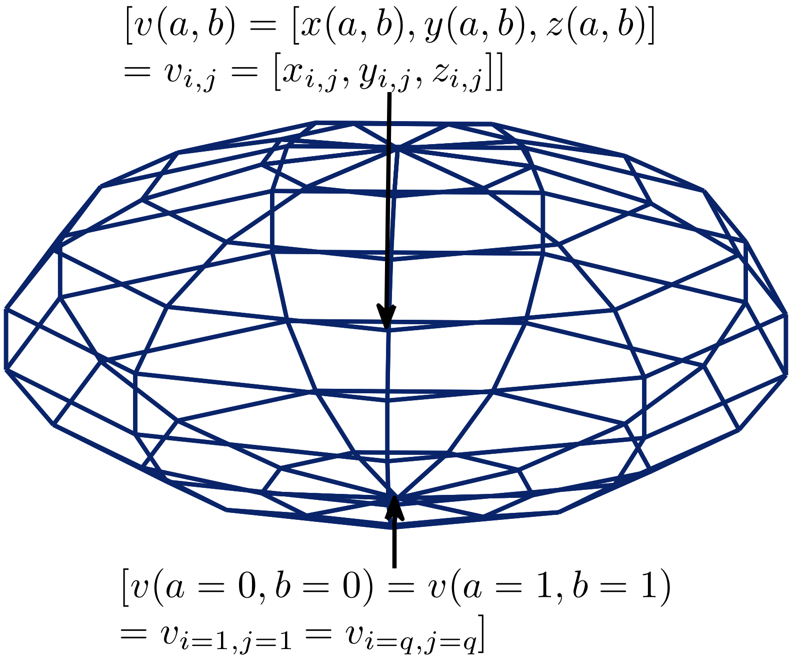

$[v(a,b) = [x(a,b), y(a,b), z(a,b)]$
$= v_{i,j} = [x_{i,j}, y_{i,j}, z_{i,j}]]$
$[v(a=0, b=0) = v(a=1, b=1)$
$= v_{i=1,j=1} = v_{i=q,j=q}]$

Fig. 2.2: A parametric active surface is represented using a random field $\mathbf{v} = \{v(a,b)\}, a, b \in [0,1]$ in the continuous domain or $\mathbf{v} = \{v_{i,j}\}, i, j \in [1,q]$ in the discrete domain. The variables a and b parameterize the 3D surface using a 2D regular mesh grid of size $q \times q$, i and j are discrete indices, and x, y and z are the components of the random vector v. Such a grid based surface representation scheme is widely used in computer vision literature to model 3D surfaces.

2.2.2 Internal Energy of the Model

The internal energy $\mathcal{E}_{int}(\mathbf{v})$, representing the prior, is a weighted sum of elastic and thin-plate energies. The internal energy $\mathcal{E}_{int}(\mathbf{v})$ of the deformable model is written as

$$\mathcal{E}_{int}(\mathbf{v}) = \int\limits_{s=0}^{1} \left(\underbrace{\alpha \left| \frac{\partial v(s)}{\partial s} \right|^2}_{elastic} + \underbrace{\beta \left| \frac{\partial^2 v(s)}{\partial s^2} \right|^2}_{thin-plate} \right) ds \tag{2.7}$$

in 2D and

$$\mathcal{E}_{int}(\mathbf{v}) = \int\limits_{a=0}^{1}\int\limits_{b=0}^{1} \left(\mathcal{E}_{elastic}\left(v(a,b)\right) + \mathcal{E}_{thin-plate}\left(v(a,b)\right)\right) da\; db \tag{2.8}$$

in 3D, where

$$\mathcal{E}_{elastic}\left(v(a,b)\right) = \alpha_1 \left|\frac{\partial v(a,b)}{\partial a}\right|^2 + \alpha_2 \left|\frac{\partial v(a,b)}{\partial b}\right|^2 \tag{2.9}$$

and

$$\mathcal{E}_{thin-plate}\left(v(a,b)\right) = \beta_1 \left|\frac{\partial^2 v(a,b)}{\partial a^2}\right|^2 + \beta_2 \left|\frac{\partial^2 v(a,b)}{\partial b^2}\right|^2 + 2\beta_3 \left|\frac{\partial}{\partial a}\left(\frac{\partial v(a,b)}{\partial b}\right)\right|^2 \tag{2.10}$$

are the elastic and thin-plate energies of the deformable model in 3D. The parameters α, α_1, α_2 and β, β_1, β_2, β_3 are weight factors for the membrane and thin-plate energy respectively. Furthermore, while these parameters may be a function of spatial location, they are typically considered as constants in practice.

2.2.3 External Energy of the Model

The second term of (2.6), the external energy functional $\mathcal{E}_{ext}(\mathbf{v})$, is computed by integrating a potential energy function $\mathcal{P}$ along the deformable boundary $\mathbf{v}$. The external energy functional $\mathcal{E}_{ext}(\mathbf{v})$ can be expressed as

$$\mathcal{E}_{ext}(\mathbf{v}) = \int\limits_{s=0}^{1} \mathcal{P}\left(v(s)\right) ds \tag{2.11}$$

in 2D and

$$\mathcal{E}_{ext}(\mathbf{v}) = \int\limits_{a=0}^{1}\int\limits_{b=0}^{1} \mathcal{P}\left(v(a,b)\right) da\; db \tag{2.12}$$

in 3D. Typically, the parametric deformable model attracts the initial solution $\mathbf{v} = \mathbf{v}_0$ towards the image boundary by enforcing a smaller potential in those regions. Therefore, the potential energy $\mathcal{P}$ representing the measurement (observation) is made a function of the magnitude of the image gradient $\mathbf{g}$ such that

$$\mathcal{P} = -\gamma(\mathbf{g}(\mathcal{I}))^2, \tag{2.13}$$

where

$$\mathbf{g}(\mathcal{I}) = (|\nabla \mathcal{I}|)\,, \tag{2.14}$$

or

$$\mathbf{g}(\mathcal{I}) = (|\nabla G_\sigma * \mathcal{I}|)\,, \tag{2.15}$$

for some gradient ∇ of Gaussian (G_σ) with bandwidth σ, where $\mathcal{I}$ is the input 2D/3D image, γ is the weight factor for the potential energy and "$*$" is the convolution operator.

2.2.4 Deformable Model Evolution

Given the formulation for the internal and external energy, the next goal is to capture the boundary of a 2D/3D object by minimizing the total energy of (2.6) using calculus of variations. However, no closed form solution to (2.6) is known. Therefore, iterative variational methods are employed in practice to find the steady state solutions of (2.6). The deformable model $\mathbf{v}$ that minimizes (2.6), also satisfies the Euler-Lagrangian equation [20–22]

$$\frac{\partial}{\partial s}\left(\alpha \frac{\partial \mathbf{v}}{\partial s}\right) - \frac{\partial^2}{\partial s^2}\left(\beta \frac{\partial^2 \mathbf{v}}{\partial s^2}\right) - \nabla \mathcal{E}_{ext}\left(\mathbf{v}\right) = 0 \tag{2.16}$$

in 2D and

$$\begin{aligned}\frac{\partial}{\partial a}\left(\alpha_1 \frac{\partial \mathbf{v}}{\partial a}\right) + \frac{\partial}{\partial b}\left(\alpha_2 \frac{\partial \mathbf{v}}{\partial b}\right) - \frac{\partial^2}{\partial a^2}\left(\beta_1 \frac{\partial^2 \mathbf{v}}{\partial a^2}\right) - \frac{\partial^2}{\partial b^2}\left(\beta_2 \frac{\partial^2 \mathbf{v}}{\partial b^2}\right)& \\ - 2\frac{\partial}{\partial a}\left(\frac{\partial}{\partial b}\left(\beta_3 \frac{\partial}{\partial a}\left(\frac{\partial \mathbf{v}}{\partial b}\right)\right)\right) - \nabla \mathcal{E}_{ext}\left(\mathbf{v}\right) = 0&\end{aligned} \tag{2.17}$$

in 3D. To study the dynamics of the deformable model, (2.16) or (2.17) can be viewed as a force balance equation [21, 22]

$$\mathcal{F}_{int}(\mathbf{v}) + \mathcal{F}_{ext}(\mathbf{v}) = 0, \tag{2.18}$$

where the internal force is expressed as:

$$\mathcal{F}_{int}(\mathbf{v}) = \frac{\partial}{\partial s}\left(\alpha \frac{\partial \mathbf{v}}{\partial s}\right) - \frac{\partial^2}{\partial s^2}\left(\alpha \frac{\partial^2 \mathbf{v}}{\partial s^2}\right) \tag{2.19}$$

in 2D and

$$\begin{aligned}\mathcal{F}_{int}(\mathbf{v}) =& \frac{\partial}{\partial a}\left(\alpha_1 \frac{\partial \mathbf{v}}{\partial a}\right) + \frac{\partial}{\partial b}\left(\alpha_2 \frac{\partial \mathbf{v}}{\partial b}\right) - \frac{\partial^2}{\partial a^2}\left(\beta_1 \frac{\partial^2 \mathbf{v}}{\partial a^2}\right) \\ & - \frac{\partial^2}{\partial b^2}\left(\beta_2 \frac{\partial^2 \mathbf{v}}{\partial b^2}\right) - 2\frac{\partial}{\partial a}\left(\frac{\partial}{\partial b}\left(\beta_3 \frac{\partial}{\partial a}\left(\frac{\partial \mathbf{v}}{\partial b}\right)\right)\right)\end{aligned} \tag{2.20}$$

in 3D. The external force in 2D and 3D are given by

$$\mathcal{F}_{ext}(\mathbf{v}) = -\nabla \mathcal{E}_{ext}\left(\mathbf{v}\right). \tag{2.21}$$

The steady state solution of (2.16) and (2.17) is obtained by treating the deformable model $\mathbf{v}$ as a function of time t and parametric space (s) or (a, b), i.e., $\mathbf{v}_t(s)$ in 2D and $\mathbf{v}_t(a, b)$ in 3D. Assuming a small amount of drift $\varepsilon \frac{\partial \mathbf{v}}{\partial t}$ is required to make the internal and external force equal, (2.18) can be re-expressed as

$$\mathcal{F}_{int}(\mathbf{v}) + \mathcal{F}_{ext}(\mathbf{v}) = \varepsilon \frac{\partial \mathbf{v}}{\partial t}, \tag{2.22}$$

where the term ε equalizes the units of right- and left-hand side. The equation (2.22) can be interpreted as a form of gradient descent optimization technique [20–22]. The numerical solution technique for (2.22) is provided next.

2.2.5 Numerical Solution Technique

Each component of (2.22) in 3D can be written as [20]:

$$\mathcal{F}_{int}(\mathbf{x}) + \mathcal{F}_{ext}(\mathbf{x}) = \varepsilon \frac{\partial \mathbf{x}}{\partial t}, \tag{2.23}$$

$$\mathcal{F}_{int}(\mathbf{y}) + \mathcal{F}_{ext}(\mathbf{y}) = \varepsilon \frac{\partial \mathbf{y}}{\partial t}, \tag{2.24}$$

and

$$\mathcal{F}_{int}(\mathbf{z}) + \mathcal{F}_{ext}(\mathbf{z}) = \varepsilon \frac{\partial \mathbf{z}}{\partial t}. \tag{2.25}$$

Using a discrete formulation, the internal force for each component of $\mathbf{v}$ in 3D is given by

$$\mathcal{F}_{int}(\mathbf{x}) = \mathcal{A}\mathbf{x}, \tag{2.26}$$

$$\mathcal{F}_{int}(\mathbf{y}) = \mathcal{A}\mathbf{y}, \tag{2.27}$$

and

$$\mathcal{F}_{int}(\mathbf{z}) = \mathcal{A}\mathbf{z}, \tag{2.28}$$

where the penta-diagonal banded matrix $\mathcal{A}$ represents the internal constraint. Employing a finite difference technique, the solution for $\mathbf{x}$ can be derived as

$$\varepsilon \frac{(\mathbf{x}_t - \mathbf{x}_{t-1})}{\delta t} = \mathcal{A}\mathbf{x}_t + \mathcal{F}_{ext}(\mathbf{x}_t), \tag{2.29}$$

$$\mathbf{x}_t = (I - \varrho A)^{-1} \left(\mathbf{x}_{t-1} + \varrho \mathcal{F}_{ext}(\mathbf{x}_{t-1}) \right), \tag{2.30}$$

where $\varrho = \frac{\delta t}{\varepsilon}$. The solutions for $\mathbf{y}_t$ and $\mathbf{z}_t$ can be derived in a similar way. For a deformable model in 2D there is no z component.

For the sake of completeness and to remind the reader that the parametric and non-parametric deformable models are two separate concepts, a brief overview of the non-parametric deformable model is provided next.

2.3 Non-Parametric Deformable Model

A fundamentally separate deformable model approach is formulated in geometric terms, such as the geometric active contour/surface (GAC) of Malladi et al. [37] and Caselles et al. [38]. The contour is described in the level-set framework of Osher et al. [71], which allows curve splitting and merging to be handled more naturally than with a spline. In level set formulation, the geometric deformable model $\mathbf{v}$ is represented by the zero level set

$$\mathbf{v}(t) = \{(x, y) | \phi(t, x, y) = 0\} \tag{2.31}$$

in 2D and

$$\mathbf{v}(t) = \{(x, y, z) | \phi(t, x, y, z) = 0\} \tag{2.32}$$

in 3D of a level set function ϕ. The evolution of the level set function ϕ can be expressed as

$$\frac{\partial \phi}{\partial t} + \mathcal{Y} \left| \nabla \phi \right| = 0, \tag{2.33}$$

where the speed function $\mathcal{Y}$ is derived from the observed data and the level set function ϕ. Such data dependent speed functions cause a boundary leakage problem. To avert this problem, Caselles et al. [35] and Siddiqi et al. [46] reformulate the level-set framework by introducing gradient-weighted length and area terms to the total energy.

The traditional solution techniques [37] to (2.33) generate shocks [53], resulting in very sharp and flat deformable models during the evolution process. To overcome these issues, the level set function is re-initialized periodically to a signed distance function. The re-initialization step of the level set formulation is a computationally expensive task. To avoid the re-initialization step of the traditional level set formulation, Li et al. [53] proposed a new variational formulation which forces the level set function close to a signed distance function.

To address the local minima issues of the level set based solution techniques, Cohen et al. [72] proposed a minimal path approach to find the global minima associated with the active contour. Mumford-Shah [73] and Chane-Vese [39, 40] developed another region based active contour model which can be solved in a level set [71] framework. The Chane-Vese model uses regional statistics as its stopping criterion, and so works efficiently even if there are very weak edges.

Recently, Sundaramoorthi et al. [74] have identified the problems associated with geometric active contour energy minimization techniques using Riemannian space. They have reformulated a generic geometric active contour by redefining the notion of gradient using a Sobolev type inner product, while using the level set methods as the evolution framework. To reduce the computational complexity of the Chane-Vese model [39], Bresson et al. [36] proposed a new global optimization method, the fast active contour (FAC), which is similar to the Chane-Vese model but with a dual optimization method. Eric et al. [44] tried a hybrid approach to improve the convergence speed and initialization robustness of level set based active contour by combining k-means clustering with the level set framework.

Recently, a new category of deformable models has been proposed, motivated by the laws of physics. Examples include the gravitational force active contour model [29], the charged particle model active contour [43] (CPM), and the magneto-static active contour model (MAC) [64]. The CPM considers each pixel to be a charged particle attracted by electric fields generated from the image gradients, and claims to reduce snake initialization sensitivity. Xie et al. [64] claim that the MAC is more effective at capturing complex boundaries than the CPM. However, the computational and algorithmic complexities of the MAC are relatively high.

2.4 Limitations of Deformable Models

The traditional parametric deformable model has six important limitations:

1. initialization sensitivity,

2. outlier sensitivity,

3. high curvature sensitivity,
4. poor scalability,
5. parameter tuning,
6. computational burden,
7. the non-parametric deformable models are mostly robust to the first five issues, but these models have another issue that is the inability to identify the boundary of broken objects.

Next, how these seven problems are addressed in current literature is discussed.

2.4.1 Initialization Sensitivity

The classical parametric deformable model is sensitive to the size and position of the initial boundary, normally manually set, and often fails to converge or takes inordinately long to converge. To avoid these issues, Cohen et al. [23], Xu et al. [21] and Li et al. [22] proposed the balloon force (BF), gradient vector flow (GVF) force and vector field convolution (VFC) force respectively to extend the influence of the external force. However, these methods still suffer from lower convergence rates and increased noise sensitivity. Recently, a Poisson inverse gradient (PIG) [54] based automatic deformable model initialization technique is proposed to reduce initialization sensitivity of the traditional deformable models.

Since the proposed approach's segmentation accuracy as a function of initialization is compared with the modified external force based methods, such as BF [23], GVF [21], VFC [22] and PIG [54] techniques in Sections 4.4, Sections 5.5 and Sections 6.6, a description of these compared methods are provided next.

Balloon Force (BF)

As shown in (2.22), in the absence of the external force $\mathcal{F}_{ext}$, the vertices of the deformable model move by the influence of the internal force $\mathcal{F}_{int}$. However, the sole purpose of the internal force is to keep the deformable boundary smooth, not to provide external thrust to the vertices of the deformable model. As a result, the vertices of the deformable model move very slowly for those cases where the deformable model is initialized far from the true boundaries.

To accelerate the convergence rate and to avoid the local minima, Cohen et al. [23] introduced a dynamic balloon force:

$$\mathcal{F}_{balloon} = \wp_1 \mathbf{n} - \wp_2 \frac{\nabla \mathcal{P}}{|\nabla \mathcal{P}|}, \tag{2.34}$$

also known as inflation/deflation force. This balloon force drives the vertices of the deformable model towards the desired boundaries in the absence of the external force. The terms $\wp_1$ and $\wp_2$ are two constants. To ensure that the balloon force is not overcoming an edge point, the value of $\wp_2$ is chosen to be slightly more than the value of $\wp_1$. The sign of $\wp_1$ determines the direction of the balloon force.

Given this formulation of the balloon force, the desired boundary is obtained using the evolution equation

$$\mathcal{F}_{int}(\mathbf{v}) + \mathcal{F}_{ext}(\mathbf{v}) + \mathcal{F}_{balloon}(\mathbf{v}) = \varepsilon \frac{\partial \mathbf{v}}{\partial t}. \tag{2.35}$$

Although the balloon based pressure force can avoid spurious edges, the balloon force creates boundary leakage when the boundaries are broken. Determining the direction of the balloon force is another common limitation of balloon force based deformable models.

Gradient Vector Flow (GVF)

To increase the capture range while addressing the issues of the balloon force deformable model, Xu et al. [21] proposed a gradient vector flow technique (GVF) to diffuse the gradient throughout the image. The gradient flow vector

$$\mathcal{F}_{GVF} = [\mathcal{F}_{GVF_x}, \mathcal{F}_{GVF_y}, \mathcal{F}_{GVF_z}], \tag{2.36}$$

is computed by minimizing the energy functional

$$E_{GVF} = \int\int\int \left(\underbrace{\mathcal{RT}}_{regularization} + \underbrace{\mathcal{DFT}}_{data-fidelity} \right) dxdydz, \tag{2.37}$$

where the regularization term is expressed by

$$\mathcal{RT} = \mu \left(|\nabla \mathcal{F}_{GVF_x}|^2 + \left|\mathcal{F}_{GVF_y}\right|^2 \right) \tag{2.38}$$

in 2D or

$$\mathcal{RT} = \mu \left(\left|\nabla \mathcal{F}_{GVF_x}\right|^2 + \left|\nabla \mathcal{F}_{GVF_y}\right|^2 + \left|\nabla \mathcal{F}_{GVF_z}\right|^2 \right) \tag{2.39}$$

in 3D, and the data fidelity term is expressed by

$$\mathcal{DFT} = \left|\nabla \mathbf{g}\right|^2 \left(\mathcal{F}_{GVF} - \left|\nabla \mathbf{g}\right|\right)^2 \tag{2.40}$$

in 2D and 3D, and $\mathbf{g}$ is the edge map derived from the observed image $\mathcal{I}$, and the regularization constant μ controls the smoothness of the gradient vector field. Given the gradient vector flow field $\mathcal{F}_{GVF}$, the evolution of the deformable model can be written

$$\mathcal{F}_{int}(\mathbf{v}) + \mathcal{F}_{GVF}(\mathbf{v}) = \varepsilon \frac{\partial \mathbf{v}}{\partial t}. \tag{2.41}$$

The GVF technique has the following drawbacks [22]: high computational cost, noise sensitivity, parameter sensitivity and ambiguity between capture range and parameters.

Vector Field Convolution (VFC)

To address the issues associated with the gradient vector flow technique, Li et al. [22] introduced a vector field convolution (VFC) based external force $\mathcal{F}_{VFC}$, where

$$\mathcal{F}_{VFC} = \mathbf{g} * \mathcal{K} \tag{2.42}$$

is estimated by convolving a vector field kernel $\mathcal{K}$ with an image edge map $\mathbf{g}$. The vector field kernel is given by

$$\mathcal{K} = \mathbf{m} \times \mathbf{n}, \tag{2.43}$$

where $\mathbf{m}$ and $\mathbf{n}$ are the magnitude and direction of the vector field. Several formulations for $\mathbf{m}$ and $\mathbf{n}$ are discussed in [22]. The evolution equation using vector field convolution based external force is given by

$$\mathcal{F}_{int}(\mathbf{v}) + \mathcal{F}_{VFC}(\mathbf{v}) = \varepsilon \frac{\partial \mathbf{v}}{\partial t}. \tag{2.44}$$

Although VFC is faster and less sensitive to noise when compared to GVF, the initialization sensitivity of VFC in the presence of outliers still remains an issue.

2.4.2 Outlier Sensitivity

Most deformable model based solutions are susceptible to outliers (local minima), many of which are created by the presence of noise, background clutter and other invalid edge-like features. Several techniques [21,22,75–78] have been proposed to improve the performance of deformable models in the presence of outliers. Some of these methods employ dedicated outlier detection techniques such as:

1. validation gates [76] to reduce the search space,
2. geometric and dynamic constraints [77, 78] to reduce shape and motion variability,
3. robust estimators [79, 80] to reduce the effect of outliers on final boudary/shape estimation,
4. probabilistic data association filters and joint probabilistic data association filters [81] to keep the track of the object's shape and motion, and
5. stroke grouping and expectation maximization (EM) [75] to avoid the convergence of the deformable model towards noise and background clutter.

However these methods to handle outliers are very time consuming, parameter dependent and insensitive to high curvature boundaries. Further, these issues become more prominent for active surface identification.

Alternatively, denoising models [36], multi-scale techniques and pre-smoothing filters [20–22] have been incorporated into the active contour formulation. For example, as shown in (2.15), to overcome noise and background clutter the external forces are usually computed by first convolving the observed images using a smoothing kernel [20–22]. However, typically the smoothing kernel smoothes the object boundaries along with the noise and background clutter.

2.4.3 Inability to Capture High Curvature Regions

A key challenge in prior-model based parametric deformable models is capturing high curvature (concave and convex) regions. Because the internal force of most prior models penalize curvature, the traditional external force is not able to pull the deformable model towards concave regions.

To enable the deformable model to converge towards concave regions, the capture range of the external force has been increased using distance force [28], balloon force [23], gradient vector flow force [21] and vector field convolution force [22]. However, as discussed in Subsection. 2.4.1, these methods have their own advantages and disadvantages, and typically are not able to attract the deformable boundary towards very high curvature regions due to the internal stiffness constraints.

Alternative methods [82, 83] have been proposed to capture high curvature regions by defining another external force which weakens the internal force near high curvature regions and strengthens the internal force near smooth regions. However, discriminating true high curvature and false high curvature generated due to outliers is a challenging task. Hence, decreased penalties lead to increased outlier sensitivity.

To nullify the effect of the internal constraints near high curvature regions in a clever way, Wong et al. [32] proposed a segmented snake, which splits the snake into a number of separated regions based on curvature, followed by later merging. Analogous to a weakened prior, the split/merge snake is similarly sensitive to noise.

2.4.4 Parameter Tuning

There is a delicate balance required between the energy parameters α, α_1, α_2, β, β_1, β_2, β_3 and γ, therefore parameter tuning is a common problem with deformable models [20–22]. The selection of appropriate parameters is generally tedious and image dependent, and the parameter sensitivity precludes the use of such algorithms in certain applications.

2.4.5 Computational Burden

Parametric deformable models use iterative techniques to find the desired solution. Therefore, the computational load of parametric deformable models primarily depend on three factors:

1. the number of vertices,

2. the convergence rate and the numerical stability, and

3. the termination criteria.

Number of Vertices

The computational and storage burden of one iteration of a deformable model is primarily a function of the number of deformable vertices. To model a large size 2D object or a moderate size 3D object a large number of deformable model vertices are required, thus implying a huge computational and storage burden.

Convergence Rate and Numerical Stability

The convergence rate and the numerical stability of the deformable model mainly depend upon the numerical stiffness of (2.6). In many cases, (2.6) is numerically stiff, which means the iterative solution of (2.6) tends to be slow and worsens if the active contour is initialized far from the true solution, implying a limited capture range.

To increase the convergence speed and capture range, most variations on the parametric deformable model have concentrated on altering the external energy, such as the pressure based balloon force [23], distance transformed image gradient [28, 29], gradient vector flow [21] and vector field convolution [22] of image gradient.

To improve the numerical stability and the convergence rate of the deformable model based solution techniques, Amini et al. [52, 84] first proposed the use of dynamic programming to perform the active contour energy optimization, which has later been used more broadly [1, 85–87]. Williams and Shah [88] introduced a fast greedy approach to find global minima of energy functionals.

Because completely unsupervised segmentations based on dynamic programming can converge to unwanted solutions, Mortensen et al. [9] first introduced intelligent scissors – a user-interactive dynamic programming based graph search method to locate exact boundaries. Further, GrabCut [89], Lazy snapping [8] and Bayesian matting [90] are introduced for user-interactive image segmentation as well.

Termination Criteria

The deformable model evolves iteratively to minimize some energy functional, ideally stopping once the global minimum is reached. Because of noise and image clutter, the energy functional is not smooth, causing algorithms to be either trapped or delayed for many iterations at local minima. To have a criterion by which the iterative solver is halted is necessary, or global optimizers need to be developed such as those based on dynamic programming [9, 52], simulated annealing [87], or graph search methods [89]. Leymarie et al. [27] suggested a better termination criterion for deformable model with detailed analysis of various parametric snake models for tracking applications.

Both the iterative solvers and global optimizers are slow to converge. Although deformable models might otherwise be a promising approach for image segmentation and tracking, the slow rate of convergence precludes existing deformable models from being considered in real-time applications.

2.4.6 Poor Scalability

Current deformable models [21, 22, 54] in 3D are direct extensions of 2D active contours to 3D active surfaces. However, such a direct extension from 2D to 3D is not always feasible due to the following three reasons.

1. In many situations, to represent the 3-D surface of an object parametrically using two parameters a and b is difficult.
2. Even for simple objects, such as a sphere, there is a one-to-many mapping from the 3D surface to the 2D parametric space. Also, there is the possibility of many-to-one mappings for complex 3D objects.
3. Handling boundary conditions in 3D is a challenging task.

Alternatively, a conforming triangular mesh [34, 91–93] based deformable models have been proposed for surface identification task. However, computing normal, curvature and higher order derivatives in 3D are key challenges.

2.4.7 Inability to Identify Broken Boundary

Non-parametric methods [35–37, 39] are effective at capturing high curvature regions and are initialization independent. However, these methods are not able to identify the boundary of a single broken object. Also, the computational load of non-parametric methods are very high as compared to parametric methods. The computational load of non-parametric methods are related to the number of pixels/voxels of an image, where as the computational load of parametric methods depend upon the number of deformable vertices. The number of deformable model vertices are much fewer compared to the number of pixels/voxels of an image.

2.5 Objectives

The aim of the thesis is to develop a generic approach to identify the boundary of a 2D/3D object. The new method (DDM) has the following six improvements compared published parametric approaches.

1. Robustness to noise and background clutter.
2. Able to capture high curvature boundary.
3. Robust to parameter tuning.
4. Insensitivity to initialization.
5. Fast convergence rate.
6. Scalable to multiple dimensions.

Compared to non-parametric methods, DDM has a higher convergence rate and less sensitivity to noise. To achieve these objectives, a decoupled deformable model (DDM) is formulated in Chapter 3 to minimize the total energy of the traditional deformable model 2.1.

First, Chapter 4 develops a decoupled active contour (DAC), a DDM in 2D to accurately and rapidly identify the boundary of a 2D object. The performance of DAC in achieving the first five objectives is evaluated against three published methods using a wide range of 2D images. Second, Chapter 5 describes a fast decoupled active contour or FDAC to improve the the computational efficiency and scalability of DAC. Finally, Chapter 6 proposes a decoupled active surface (DAS), an extension of FDAC from 2D to 3D to identify the surface of a volumetric 3D object or the tunnel of a moving 2D object.

Chapter 3

Problem Formulation

Statistical methods based on a Bayesian formulation have been recognized as one of the key concepts for solving several computer vision problems due to their ability to integrate measured information from different sensors with the historical information about these measurements [30, 94]. As discussed in Chapter 2, the boundary identification task using deformable models can be modeled using statistical methods [30]. Therefore, this chapter proposes a non-iterative coordinate descent technique to solve the parametric deformable model energy formulation (2.6) in a Bayesian paradigm. The proposed approach aims to address the limitations of present deformable model based methods and to achieve the goals mentioned in Section 2.5.

As described in Subsection 2.2.1, the total energy $\mathcal{E}_{tot}$ of the parametric deformable model $\mathbf{v} = \{v(s)\}, s \in [0, 1]$ in 2D or $\mathbf{v} = \{v(a, b)\}, a \in [0, 1], b \in [0, 1]$ in 3D is expressed as:

$$\mathcal{E}_{tot} = \mathcal{E}_{int} + \mathcal{E}_{ext}. \tag{3.1}$$

In a Bayesian framework [30], the total energy $\mathcal{E}_{tot}$, the internal energy $\mathcal{E}_{int}$ and the external energy $\mathcal{I}$ of the traditional deformable model $\mathbf{v}$ can be expressed as the posterior $P(\mathbf{v}|\mathcal{I})$, the prior $P(\mathbf{v})$ and the measurement $P(\mathcal{I}|\mathbf{v})$ of a random field $\mathbf{v}$, respectively. Therefore, following the formulation of Gibbs random field [30,95,96] (GRF) the (3.1) can be expressed as:

$$P\left(\mathbf{v}|\mathcal{I}\right) \propto P\left(\mathbf{v}\right) P\left(\mathcal{I}|\mathbf{v}\right) \tag{3.2}$$

with

$$P\left(\mathbf{v}|\mathcal{I}\right) = \frac{1}{Z_{tot}} \exp\left(-\mathcal{E}_{tot}\right), \tag{3.3}$$

$$P\left(\mathbf{v}\right) = \frac{1}{Z_{\text{int}}} \exp\left(-\mathcal{E}_{\text{int}}\right) \tag{3.4}$$

and

$$P\left(\mathcal{I}|\mathbf{v}\right) = \frac{1}{Z_{ext}} \exp\left(-\mathcal{E}_{ext}\right), \tag{3.5}$$

where the normalization constants Z_{tot}, Z_{int} and Z_{ext} are normalization constants. Theoretically, the true converged solution $\mathbf{v^c}$ representing the boundary of a 2D/3D object can be obtained by maximizing the posterior (3.2), $P\left(\mathbf{v}|\mathcal{I}\right)$ (MAP problem):

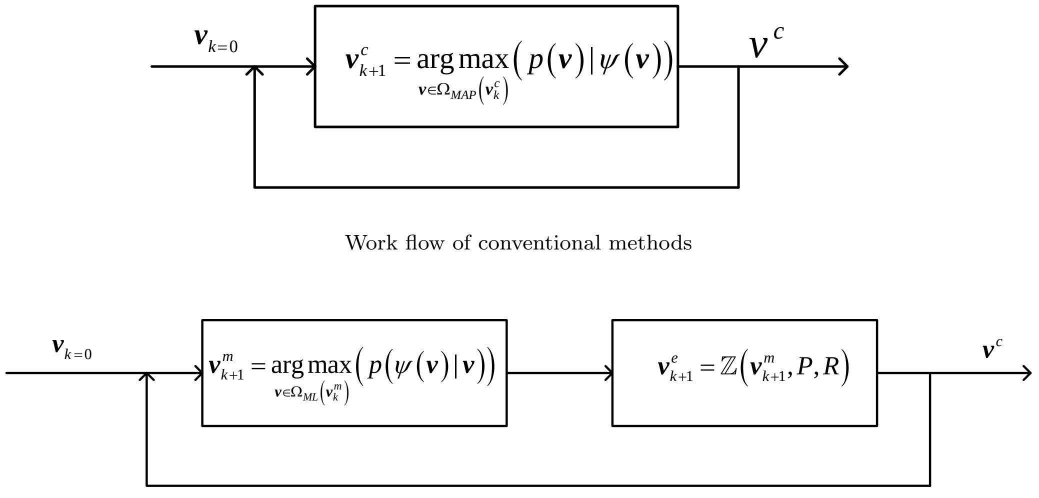
$$\mathbf{v}^c = \arg\max_{\mathbf{v}} \left(P\left(\mathbf{v}|\mathcal{I}\right)\right) \tag{3.6}$$

or minimizing (3.1), the total energy of the parametric deformable model.

However, the desired solution $\mathbf{v^c}$ of (3.6) typically lies in a complex solution space with a significant chance of being trapped in a local minimum, especially in the presence of outliers. Further, no closed form solution of (3.6) is known. Hence, usually iterative minimization techniques [20, 30, 95, 96] including simulated annealing have been proposed to obtain the minima of (3.6). For illustration, the work flow of traditional iterative minimization techniques to solve (3.6) is described in the top panel of Fig. 3.1 using a flow diagram, where variational approaches [20,36,95,96] are adopted to find the final converged solution $\mathbf{v}^c$, iteratively. Essentially, each iteration of these variational methods finds an intermediate solution in the local neighborhood of the previous solution ($\Omega_{MAP}(v^p)$). These iterative methods are inherently slow and are local in nature. Therefore, a fast, large and outlier robust method to solve the MAP problem is essential.

The complex, nonlinear nature of this MAP problem makes an optimal, non-iterative method essentially impossible, however the key insight is that the two parts (prior and measurement) of (3.6) can, in fact, be individually optimized in a non-iterative fashion within a specified search space. Although the overall algorithm remains iterative, alternating between the external and internal criteria, each iteration converges far more rapidly than the direct iterative solutions to (3.6).

Based on these ideas, this thesis proposes a decoupled deformable model (DDM) which solves this MAP problem in two separate steps.


$\boldsymbol{v}_{k=0}$
$\boldsymbol{v}_{k+1}^{c} = \underset{\boldsymbol{v}\in\Omega_{MAP}(\boldsymbol{v}_k^c)}{\arg\max}\left(p(\boldsymbol{v})\,|\,\psi(\boldsymbol{v})\right)$
V^{c}
$\boldsymbol{v}_{k=0}$
$\boldsymbol{v}_{k+1}^{m} = \underset{\boldsymbol{v}\in\Omega_{ML}(\boldsymbol{v}_k^m)}{\arg\max}\left(p(\psi(\boldsymbol{v})\,|\,\boldsymbol{v})\right)$
$\boldsymbol{v}_{k+1}^{e} = \mathbb{Z}\left(\boldsymbol{v}_{k+1}^{m}, P, R\right)$
$\boldsymbol{v}^{c}$

Work flow of conventional methods

Work flow of DDM

Fig. 3.1: Demonstration of the work flow of the conventional methods and the new decoupled deformable model (DDM). The traditional techniques use iterative local variational techniques to find a desired solution $\mathbf{v}^c$, iteratively. Where each iteration of these local techniques obtain an intermediate solution $\mathbf{v}_{k+1}$ using a local solution space $\Omega_{MAP}(\mathbf{v}^p)$ (top panel). However, each iteration of the DDM (bottom panel) separates the prior from the measurement and first, finds an approximate solution $\mathbf{v}_{k+1}^m$ using a maximum likelihood (ML) estimator from a large solution space $\Omega_{ML}(\mathbf{v}^p)$ as demonstrated in Fig. 3.2. Then, DDM enforces a predefined prior P and noise statistics R into the approximate solution $\mathbf{v}_{k+1}^m$ using a statistical estimator $\mathbb{Z}$. These two steps of DDM are iteratively performed until a desired converged solution $\mathbf{v}^c$ is obtained.

1. First, DDM applies a maximum likelihood (ML) estimator to find an approximate solution $\mathbf{v}_{k+1}^m$ in the vicinity ($\Omega_{ML}(\mathbf{v}^p)$) of the previous solution ($\mathbf{v}^p$), by ignoring the prior.

2. Second, DDM adopts a statistical data fusion technique to enforce a predefined prior into the solution obtained from the first step.

These two steps of DDM are performed iteratively to find a converged solution $\mathbf{v}^c$ as described in the bottom panel of Fig. 3.1. DDM needs several iterations to find the desired boundary of a complex and high curvature object and also to make the measured boundary less dependent upon the size of the search space.

The key differences between the conventional and the proposed approach of solving (3.6) are demonstrated in Figs. 3.1 and 3.2, and are summarized below.

1. As illustrated in Fig. 3.2, each iteration of DDM uses a large neighborhood $\Omega_{ML}(\mathbf{v}^p)$ compared the local neighborhood $\Omega_{MAP}(\mathbf{v}^p)$ of traditional methods to find an intermediate solution.

2. Each iteration of DDM uses a large ML estimator compared to the local variational approach of the conventional techniques to find an approximate intermediate solution.

3. The ML solution space of DDM is simpler compared to the MAP solution space of the conventional methods, as a result DDM is less likely to be trapped to a local minimum compared to the conventional approaches.

4. Since, DDM uses a larger search space $\Omega_{ML}(\mathbf{v}^p)$ (Fig. 3.2) compared to the conventional techniques, therefore, DDM is anticipated to take significantly fewer iterations than the conventional techniques to converge to a desired solution.

Following the concepts of DDM, novel 2D/3D boundary identification methods are developed in subsequent chapters.

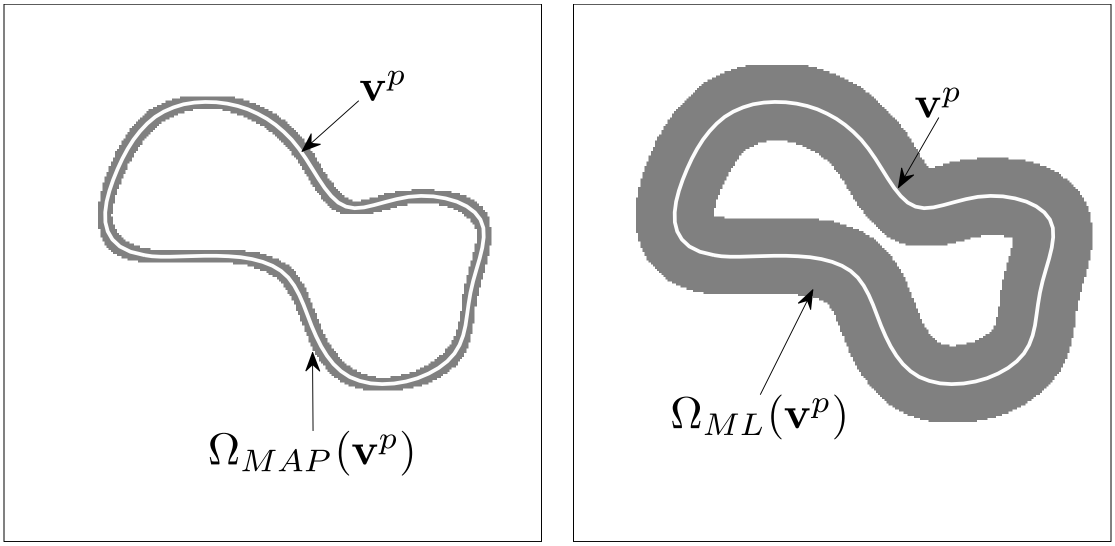


Fig. 3.2: Search space of the conventional approaches and DDM. The local (left, $\Omega_{MAP}(\mathbf{v}^p)$) and the large (right, $\Omega_{ML}(\mathbf{v}^p)$) search space of one iteration of conventional methods and the proposed DDM are demonstrated. The search area (grey) is defined around a previous solution (white curve) $\mathbf{v}^p$ for both conventional approaches and DDM. To accelerate the convergence rate and to avoid local minima, DDM uses a wider search area and a ML estimator compared to the local search space and local variational techniques of conventional approaches.

Chapter 4

Decoupled Active Contour (2D)

Finding a method to consistently identify the accurate boundary of a 2D complex object in the presence of noise and background clutter has been a challenging problem [75, 79], and current boundary extraction techniques proposed for addressing this challenge have shown a number of drawbacks as shown in (3.5) (Chapter 2.4). This chapter proposes and implements the decoupled active contour (DAC), a DDM in 2D designed to tackle the limitations of existing boundary identification approaches described in Section 2.4. By addressing these limitations, decoupled active contour aims to achieve the first five objectives specified in Section 2.5.

To validate the claims made regarding DAC in Section 2.5, the performance of DAC is compared with three state-of-the-art segmentation algorithms [20,21,40] using three testing strategies. The first test is a comprehensive quantitative segmentation accuracy evaluation on the Weizmann database [97] (Subsection 4.4.4). The second test is a quantitative evaluation of boundary identification accuracy and convergence speed using natural and synthetic images (Subsection 4.4.5). The third test demonstrates the capabilities of DAC through illustrative examples (Subsection 4.4.6). The experimental results show that DAC is capable of achieving a dramatic improvement over existing parametric active contour approaches across all five objectives involving high curvature, noise, parameter sensitivity, initialization and speed. Furthermore, compared to non-parametric approaches, DAC can achieve improved segmentation accuracy for single objects in the presence of noise and background clutter with lower computational cost.

To achieve these objectives and to overcome the slow convergence rate of traditional parametric active contours an effective solution technique would be desirable. This effective solver, known as DAC, is designed in this chapter following the concepts described in Chapter 3. The DAC simplifies the optimization step by first ignoring the prior and employing a maximum likelihood (ML) estimator to obtain a sub-optimal solution ($\mathbf{v}^m$) that is consistent with the measurements, second generating a non-stationary prior to satisfy high curvature (convex and concave) boundaries, and third by asserting the non-stationary prior via a Bayesian linear least squares estimator.

The DAC method consists of three steps, illustrated in Fig. 4.1.

Step 1: Measurement. The measured boundary finding problem is modeled as an HMM and a Viterbi search is used to find the solution by dynamic programming. In the absence of image noise and shape prior, the Viterbi [98] search will identify all of the strongest local boundaries. Details are developed in Section 4.1.

Step 2: Resampling. If active contour vertices are uniformly spaced on the curve, there will be an excess of vertices in areas of gentle curvature and too few vertices in areas of high curvature. To make a single algorithm work in both smooth and high-curvature portions of a curve, a non-stationary prior is essential, which is accomplished by placing more active contour vertices in high-curvature areas, weakening the prior, and fewer in smooth areas, strengthening the prior. Therefore the curvature of the Viterbi boundary is computed and importance sampled to generate non-uniform samples. Details are developed in Section 4.2.

Step 3: Statistical Estimation. The non-stationary prior constraints need to be traded-off against the strength and significance of the image gradients. The fused curve is estimated statistically, as described in Section 4.3.

4.1 Step 1: Measurement

Although a large number of techniques [20–22, 24, 37, 39, 40] for object boundary extraction have been proposed, most of these methods employ local iterative strategies to identify the converged boundary. By using local iterative strategies, these methods often converge to the wrong boundaries caused by local minima. To avoid such a scenario, DAC first

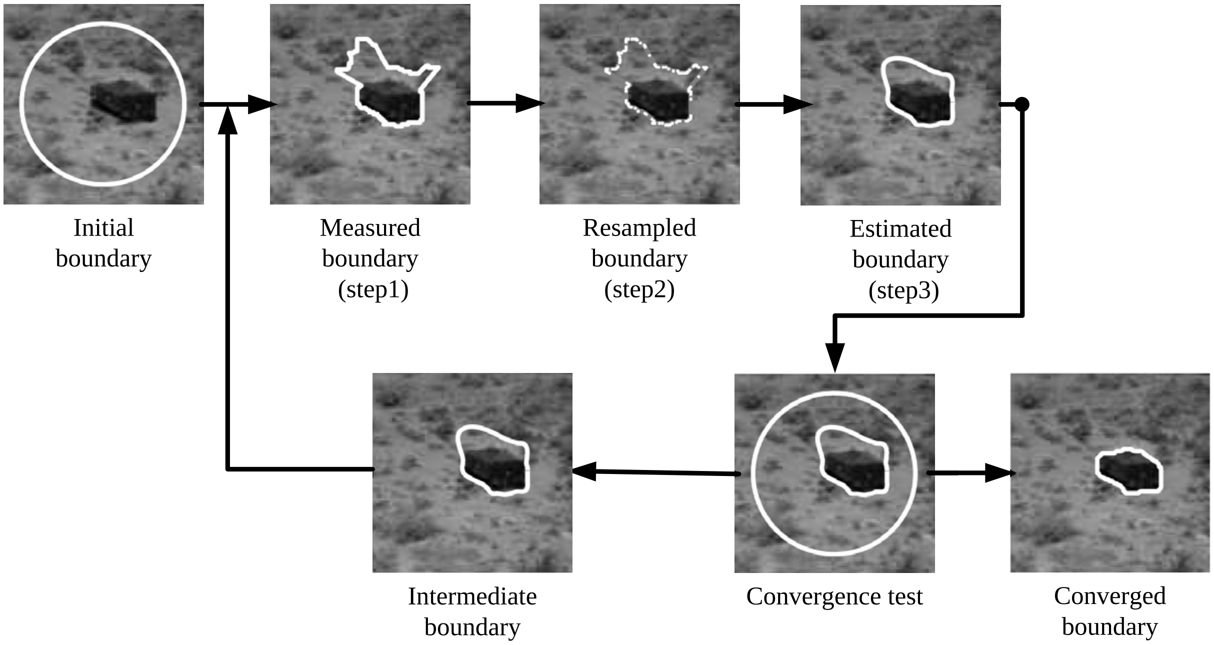

Initial boundary
Measured boundary (step1)
Resampled boundary (step2)
Estimated boundary (step3)
Intermediate boundary
Convergence test
Converged boundary

Fig. 4.1: Flow diagram of DAC. The top left panel shows the initial boundary (white circle). The three important steps of DAC: finding the measured boundary using an hidden Markov model (step 1), generation of non-stationary boundary (step 2), and statistical estimation (step 3) are shown. The stopping criterion is determined by measuring the error between the current and previous boundaries. The final converged boundary is shown in the bottom right panel.

computes a non-local approximate measured boundary by ignoring the prior. This section introduces a practical solution technique to find the measured boundary $\mathbf{v}^m$ based on the following formulation:

$$\mathbf{v}^m = \underset{\mathbf{v}}{\operatorname{argmax}} \left(P \left(\mathcal{E}_{ext} | \mathbf{v} \right) \right), \tag{4.1}$$

where $\mathbf{v}$ is a random field and $P \left(\mathcal{E}_{ext} | \mathbf{v} \right)$ is the likelihood function of the random field as described in Chapter 3. To efficiently solve 4.1, the measured boundary extraction problem is formulated as an Hidden Markov Model (HMM) on a discrete space, which allows the solution to be found using a standard Viterbi search [98]. The initial and important step involved in HMM [99] is the generation of the solution or the search space or the trellis.

The generation of the trellis can be described as follows. To precisely locate the boundary of complex high curvature objects, it is essential to carry out the three main steps of DAC for several iterations. Therefore, to avoid undoing the work done by previous iterative updates, the search space needs to be constrained to seek an optimum in the vicinity of $\mathbf{v}^p$, which denotes the solution obtained by the previous iteration. Generation of the trellis begins by discretizing the random field $\mathbf{v}$ using a collection of $q-1$ discrete straight segments with q discrete locations (the head and tail of the active contour do not have to be connected):

$$\mathbf{v}_0 = \{v_j\} = \{v(s_j) = v_{j,w_j=0} = (x_{j,0}, y_{j,0}) = (x_j, y_j)\}, \;\; j \in [1, \cdots, q], s_j \in [0, 1]. \qquad (4.2)$$

The trellis or the search space is defined with reference to the initial solution $\mathbf{v}_0$.

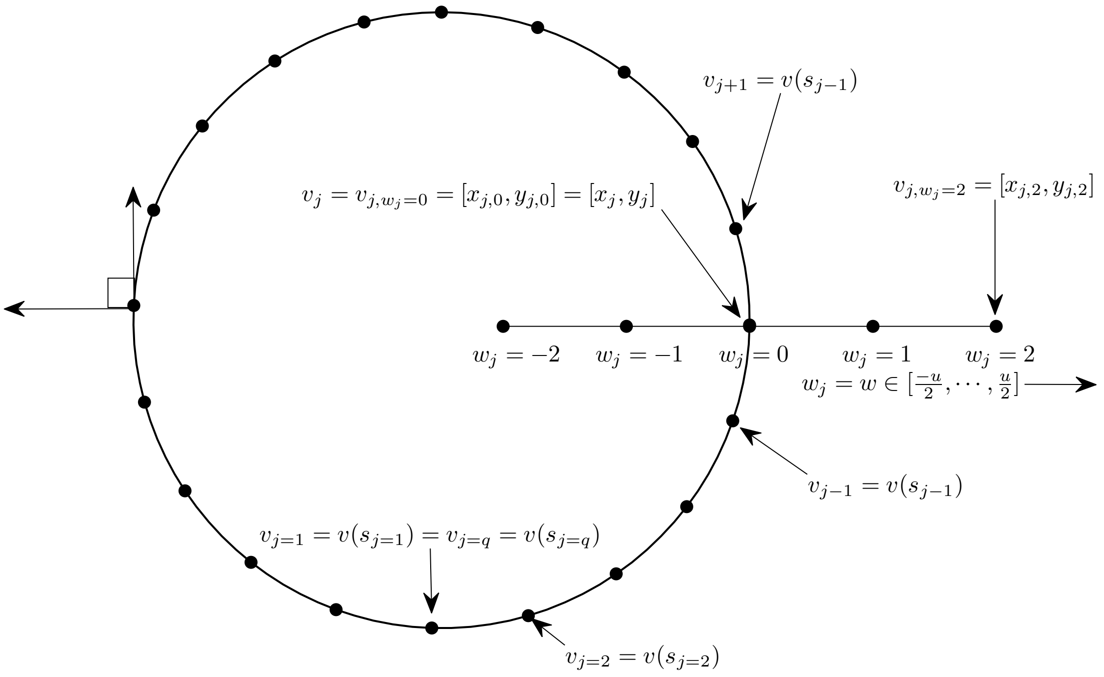

$v_{j+1} = v(s_{j-1})$
$v_j = v_{j,w_j=0} = [x_{j,0}, y_{j,0}] = [x_j, y_j]$
$v_{j,w_j=2} = [x_{j,2}, y_{j,2}]$
$w_j = -2$
$w_j = -1$
$w_j = 0$
$w_j = 1$
$w_j = 2$
$w_j = w \in [\frac{-u}{2}, \cdots, \frac{u}{2}]$
$v_{j-1} = v(s_{j-1})$
$v_{j=1} = v(s_{j=1}) = v_{j=q} = v(s_{j=q})$
$v_{j=2} = v(s_{j=2})$

Fig. 4.2: Description of DAC terminology. $s_j \in [0, 1]$ where $j \in [1, \cdots, q]$ is the normalized discretized arclength along the active contour; $v_{j|w_j=i} = v_{j,i} \in \mathbb{R}^2$ is the position of a node in (x, y) space. The jth normal is constructed at s_j, with $u + 1$ possible values (small circles) for the active contour at that normal. The index j varies along the deformable model while the index i varies along the normals. A complete trellis is shown in Fig. 4.3.

As illustrated in Fig. 4.2, at each of the q discretized deformable model locations, a set of $u + 1$ nodes are defined lying normal to the initial curve $v_{j,w_j=0}, j \in [1, \cdots, q]$. The

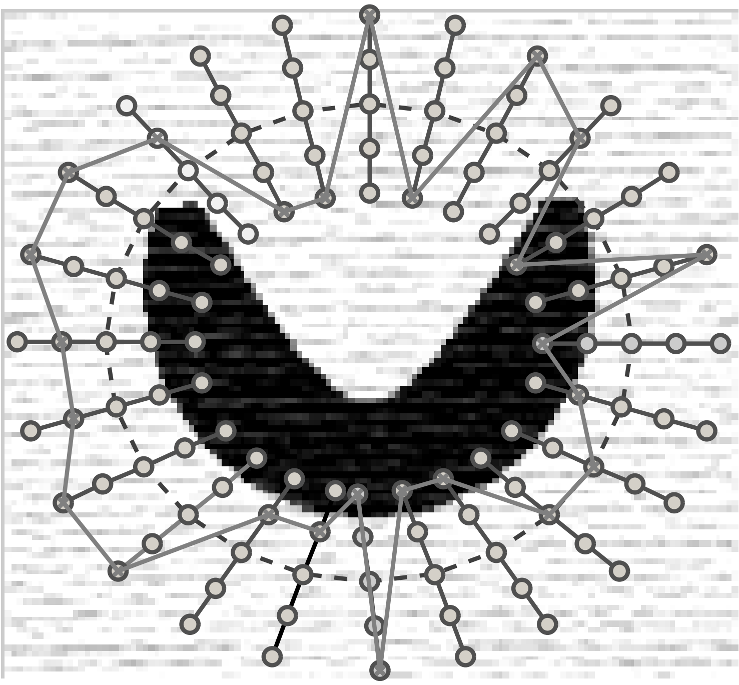


Fig. 4.3: Circular trellis (hidden Markov model (HMM)) with an initial active contour of random field ($\mathbf{v}_0$) (dotted grey circle), one possible, albeit unlikely active contour (jagged, dotted grey), normals (grey lines) and nodes (small circles along the normals). This trellis present q normals with each normal containing $u+1$ number of nodes. Therefore, in total q^{u+1} possible active contours (solutions) can be generated from this trellis using (4.4).

positions of the nodes of the trellis along the normals in 2D are computed as

$$v_{j,i} = v_{j,0} + i \times n_j, \;\; j \in [1, \cdots q], i \in [-\frac{u}{2}, \cdots, \frac{u}{2}], \;\; u \text{ is even}, \tag{4.3}$$

where $v_{j,i} \in \mathbb{R}^2$ is the position of a node in (x, y) plane. If we fix the vector w, then the positions of these nodess in a 1D manifold can be represented as

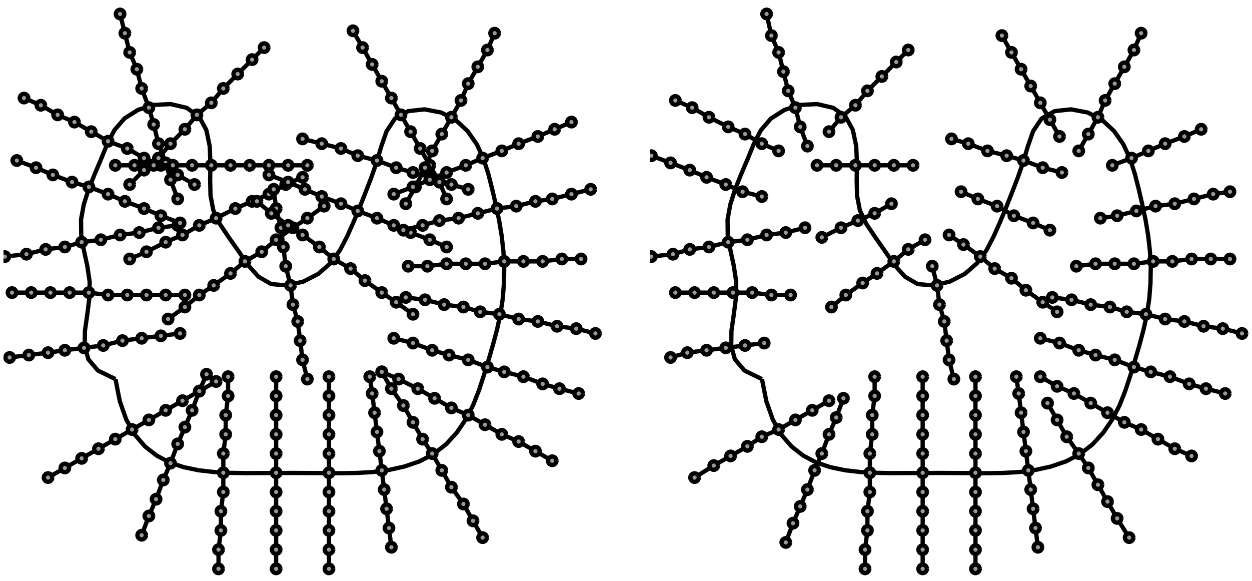
$$v_{j,w_j} = v_{j,0} + w_j \times n_j, \;\; j \in [1, \cdots q], w_j \in [-\frac{u}{2}, \cdots, \frac{u}{2}], \;\; u \text{ is even} \tag{4.4}$$

where the unit normal n_j for node j is computed as

$$n_j = [n_{x_{j,0}}, n_{y_{j,0}}] = \frac{[-\Delta y_j, \Delta x_j]}{(\Delta y_j^2 + \Delta x_j^2)^{\frac{1}{2}}}, \tag{4.5}$$

where $\Delta x_j = x_{j,0} - x_{j-1,0}$ and $\Delta y_j = y_{j,0} - y_{j-1,0}$. As presented in Fig. 4.3, q^{u+1} number of potential boundaries can be generated by varying the states $w_j = w \in [\frac{-u}{2}, \cdots, \frac{u}{2}]$ of the normals $j \in [1, \cdots, q]$.

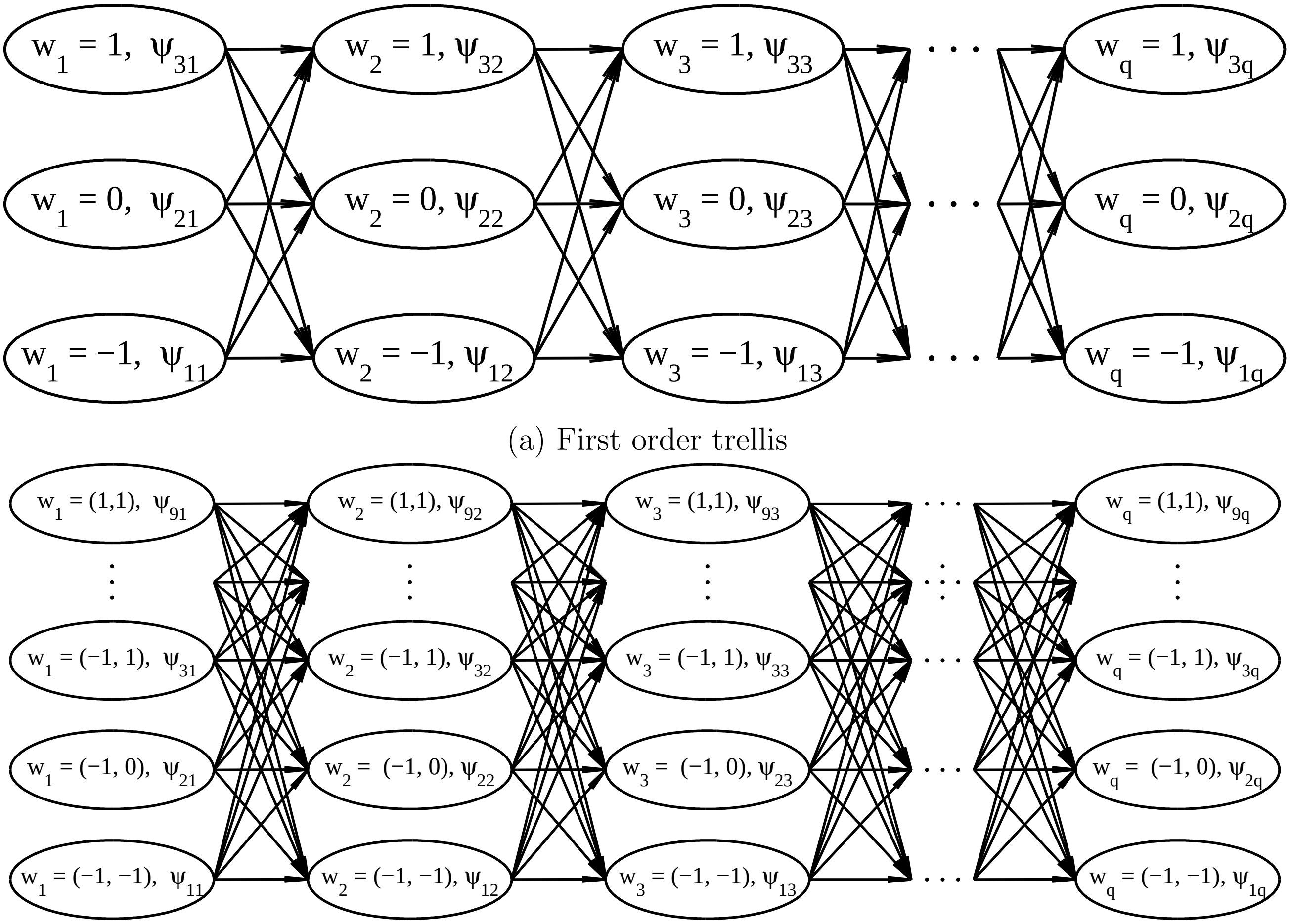
As shown in Fig. 4.4, constant-length normals lead to the possibility of a self-intersecting curve. To deal with this problem, an approach similar to the dual front propagation technique [100] is initially tried, where a medial axis growing and shrinking technique is used to generate a search space. However, such an approach is computationally expensive and fails to grow normals outwards from the curve in areas of high curvature. Instead, the length of the normals are iteratively tested and pruned until a set of non-intersecting normals is obtained. A set of non-intersecting normals corresponding to Fig. 4.4(a) is illustrated in Fig. 4.4(b).



(a) Constant length normals

(b) non-crossing normals

Fig. 4.4: Left, constant length lines (black lines with circular grey nodes) normal to the curve $\mathbf{v}$ (black) at the locations $j \in [1, \cdots, q]$. Right, non crossing pruned normals obtained from (a). Constant length normals cross with each other. As a result these normals can create loops in the measured boundary. Hence, to avoid the unwanted loops in the measured boundary the constant length normals are pruned iteratively as shown in (b).



(a) First order trellis

(a) Second order trellis

Fig. 4.5: First- and second-order trellis, where $w_1, w_2, \ldots, w_q$ are the hidden states, and $\psi(w_j) = \psi_{w_j,j}$ are the observations. The second-order trellis, bottom, has a far higher representation, storage, and computational complexity than the first-order trellis, top. In both cases, $u = 2$.

Given this set of non-intersecting normals, the exhaustive set of $(u+1)^q$ possible curves over which to optimize may be represented as the possible transitions through an ordered graph, as shown in Figs. 4.3 and 4.5. Let the jth state be expressed as

$$w_j = w \in \left[-\frac{u}{2}, \dots, \frac{u}{2}\right], \quad j \in [1, \cdots, q], \quad u \text{ is even.} \tag{4.6}$$

This means that w_j selects one of the $u+1$ evenly distributed nodes a given active contour passes through at each of the q normals. Given a sequence of states

$$\mathbf{w} = [w_1, w_2, \dots, w_q], \tag{4.7}$$

if the energy of this state sequence can be determined, then a basis to find the optimal $\mathbf{w}$ can be found. Efficient solvers of such graph problems, such as the Viterbi method [98,99], allow a cost to be associated with each state value w_j, and with each state transition $w_j \rightarrow w_{j+1}$. To assert prior models with penalties involving more than two successive state values (such as a penalty on curvature) is possible. For example a penalty on curvature (second term of (2.7)) would require *three* successive vertices. To accomplish this, one of two options are possible.

1. Let each state represent a *pair* of vertices, with the consequence that the size of the graph explodes (see bottom panel of Fig. 4.5), or
2. Let the optimization of $\mathcal{E}_{ext}$ be separated from the assertion of the prior and just use a simpler graph (see top panel of Fig. 4.5). This is the proposed approach for DAC.

The HMM will therefore be limited to simple inter-point constraints. Since the active contour prior is to be asserted later, at this point only the length of the active contour is penalized, such that the state transition probability is represented by

$$\begin{aligned} P(w_j, w_{j+1}) &= \frac{1}{\sqrt{2\pi}\sigma_s} \exp\left(-\frac{\|\Delta v(w_j, w_{j+1})\|}{2\sigma_s^2}\right) \\ &= \frac{1}{\sqrt{2\pi}\sigma_s} \exp\left(-\frac{\left\|v_{w_j} - v_{w_{j+1}}\right\|}{2\sigma_s^2}\right), \end{aligned} \tag{4.8}$$

where for simplicity

$$v_{w_j} = v_{j,w_j} = \left(x_{j,w_j}, y_{j,w_j}\right)$$

represents the vertices along the potential active contours, and therefore $\|\Delta v\|$ is the corresponding arclength between two successive vertices on the potential active contour. The term σ_s represents the spatial standard deviation and is computed from the distribution of Euclidian distances between all pairs of nodes between the jth and $(j+1)$th normals. The transition probability (4.8) discourages spiky boundary by limiting the perimeter of the deformable boundary.

An alternate transition probability might be the integral of edge map weighted arclength ($\int_{v_{w_j}}^{v_{w_{j+1}}} \psi ds$) between v_{w_j} and $v_{w_{j+1}}$, where $\psi = 1/(1+\mathbf{g})$ is the edge map of the image. This has the benefit of evaluating the edge map along each arc, rather than only at discrete vertices, however computing the edge map information along each arc is a computationally intensive process, and in practice is found to offer little to no benefit over the proposed approach based on Euclidian distance.

Next, the state probability for w_j must be related to the external energy (normally related to the image gradient $\mathbf{g}$) at the state location v_{w_j}, thus the state probability is defined as

$$P(\psi(w_j)|w_j) = \frac{1}{\sigma_{\psi(w_j)}} \exp\left(-\frac{\psi(w_j)}{\sigma_{\psi(w_j)}}\right) \tag{4.9}$$

where the measurement is defined as

$$\psi(w_j) = \frac{1}{1+\mathbf{g}(w_j)} \tag{4.10}$$

and $\sigma_{\psi(w_j)}$ is the standard deviation of the measurements along the normal j and computed locally from $\psi(w_j)$. $\mathbf{g}$ is calculated as per (2.13). The observation probability (4.9) attracts the nodes towards object boundaries.

At this point we have a first-order lattice (Fig. 4.5(a), top) with state and state-transition probabilities defined. In principle, the optimum contour can be found by solving the joint maximization

$$\max_{w_1,w_2,\cdots w_q} P\big(w_1, w_2, \cdots, w_q\big). \tag{4.11}$$

However, in practice there is no need to find the optimum, particularly because the proposed algorithm remains iterative, and because (4.11) is only one part of the complete criterion (4.12):

$$\mathbf{v}^c = \arg\max_{\mathbf{v}} \left(P\left(\mathbf{v}|\mathcal{E}_{ext}\right)\right). \tag{4.12}$$

Therefore, finding a sub-optimal path using the Viterbi algorithm [98], as described in Algorithm 1 is preferable.

Since the Viterbi search [98] is a sub-optimal algorithm, it is dependent on both the initial state w_1 and the ordering of the vertices. In principle this dependence could be removed by minimizing over all initial states and ordering, however experimentally added little in terms of accuracy, but greatly in terms of computational burden. By increasing the computational cost of the Viterbi method from $O(u^2q)$ to $O(u^2q^2)$, where typically $q = 300$, and where the Viterbi method accounts for approximately half of the total complexity of DAC, this minimization slows the method by a factor of approximately 150. Given the limited benefit and considerable cost, and given the robust convergence seen in the results, a regular Viterbi [98] method is proposed to identify the measured boundary.

The state sequence $w_j^m, j \in [1, \cdots, q]$ resulting from the Viterbi optimization gives rise to the sub-optimal curve

$$v_{j,w_j^m} = v^m(s_j) = (x_{j,w_j^m}, y_{j,w_j^m}),\ j \in [1, \cdots, q],\ \ s_j \in [1, q] \tag{4.13}$$

For notational brevity, this thesis will denote v_{j,w_j^m} as simply v_{j^m} and termed this as the measured boundary.

4.2 Step 2: Resampling

Active contours generally consider stationary elastic and thin-plate constraints to constitute the prior model of the object. Such a prior represents a poor hypothesis for complex, high curvature objects because the degree to which prior smoothness should be asserted varies with the boundary curvature.

For handling contours of high curvature, Wong et al. [32] proposed a segmented active contour model, which follows a dual optimization approach, with a rough estimate obtained using a classical active contour, followed by a recursive split and merge approach. Such an approach to capture high curvature boundaries seems to be ad hoc.

In contrast, DAC approximates the smoothness of complex boundaries using stationary elastic and rigidity constraints, but with non-stationary sampling intervals of the vertices along the active contour, which will ensure more samples (with correspondingly shorter active contour segments) near high curvature regions.

Algorithm 1 $[\mathbf{v}^m]$ = Function Viterbi($\mathbf{v}$)

```
Create the 2D trellis using $\mathbf{v}$ as the initial solution;
Compute the initial observation probability $P(\psi(w_1))$ using (4.9);
**Initialization:** j=1, $\Theta(w_1) = P(\psi(w_1)), \Gamma(w_1) = \{0\}$ and $w_1 = [-\frac{u}{2} \cdots \frac{u}{2}]$;
**while** $j \leq q$ **do**
    Compute the observation $(P(\psi(w_j)))$ and the transition probability $P(w_j, w_{j+1})$ using (4.9) and (4.8);
    $\Theta\left(w_{j+1}\right) = \max_{w_j} \left(\Theta(w_j) \times P(w_j, w_{j+1})\right) \times P(\psi(w_j))$;
    $\Gamma(w_{j+1}) = \arg\max_{w_j} \left(\Theta(w_j) P(w_j, w_{j+1})\right)$;
    $j = j + 1$;
**end while**
**Termination:** $w_q^m = \arg\max_{w_q} \left(\Theta(w_q)\right), v_q^m = (x_{q,w_q^m}, y_{q,w_q^m})$;
$j = q - 1$;
**while** $j > 2$ **do**
    $w_j^m = \Gamma(w_{j+1}^m), v_j^m = v_{j,w_j^m} = (x_{j,w_j^m}, y_{j,w_j^m})$;
    $j = j - 1$;
**end while**
```

The curvature κ of a parametric curve $v(s) = (x(s), y(s))$ is defined as the rate of change of the tangent angle and is expressed as

$$\kappa(s) = \left| \frac{\partial^2 v}{\partial s^2} \right|. \tag{4.14}$$

From [101], (4.14) can be rewritten locally as

$$\kappa\left(s\right) = \frac{\frac{\partial x(s)}{\partial s} \frac{\partial^2 y(s)}{\partial s^2} - \frac{\partial y(s)}{\partial s} \frac{\partial^2 x(s)}{\partial s^2}}{\left(\left| \frac{\partial x(s)}{\partial s} \right|^2 + \left| \frac{\partial y(s)}{\partial s} \right|^2 \right)^{\frac{3}{2}}}, \tag{4.15}$$

where the derivatives are computed numerically using forward differences. Given the Viterbi boundary $\mathbf{v}^m$, we can use discrete derivatives in (4.15) to compute the sampled curvature $\boldsymbol{\kappa} = \{\kappa_j\} = \{\kappa(s_j)\}, s_j \in [0, 1]$. For DAC, the vertices are resampled with a sampling density proportional to the negative exponential of curvature:

$$\Delta v(s_j) = \Delta v_j = \frac{l_1}{\sigma_\kappa} \exp\left(-\frac{\left|\kappa\left(s_j\right)\right|}{\sigma_\kappa} \right), \tag{4.16}$$

where $\Delta v(s_j) = \Delta v_j = \|v_j - v_{j-1}\|$ is the discrete arclength of an individual segment of curve $\mathbf{v}$ and σ_κ is the standard deviation of sampled curvature $\boldsymbol{\kappa}$.

If we fix q, the total number of vertices, then the proportionality parameter l_1 is set to preserve the arclength of the curve.

Since arbitrarily high curvature is possible, leading to arbitrarily fine discretization, for practical reasons the discretization interval needs to be limited as

$$\Delta v^m(s_j) = \begin{cases} \Delta v_{\max} & \text{if } \Delta v(s_j) > \Delta v_{\max} \\ \Delta v_{\min} & \text{if } \Delta v(s_j) < \Delta v_{\min} \\ \Delta v(s_j) & \text{otherwise,} \end{cases} \tag{4.17}$$

where

$$\frac{1}{\Delta v_{\max}} \ll q \ll \frac{1}{\Delta v_{\min}} \tag{4.18}$$

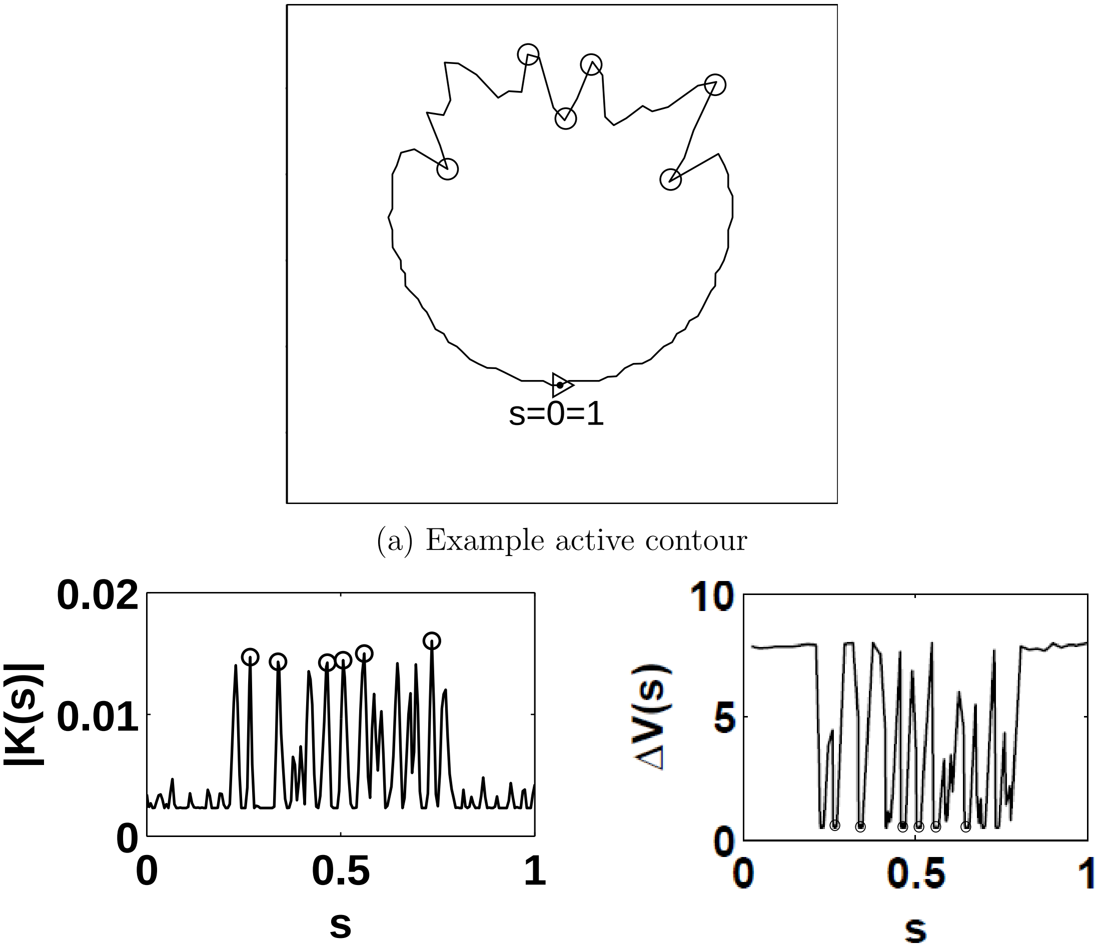
to generate q samples $\mathbf{v}^m$ having a sampling interval of $\Delta v^m(s_j)$, where the conditions (4.17) and (4.18) imposed on $\mathbf{v}^m$ are satisfied using an ad hoc trial and error algorithm.

Fig. 4.6 illustrates the resampling procedure. The inverse relationship between curvature and sampling interval are displayed in panels (b) and (c). The maximal and minimal extents of the sampling interval are determined by the user.

4.3 Step 3: Statistical Estimation

Because of the complexity of introducing all but the most trivial shape priors into the Viterbi boundary extraction, the calculation of $\mathbf{v}^m$ have not permitted the assertion of a meaningful prior. Therefore, to limit the computational complexity of the Viterbi step, but to retain the assertion of a prior, a separate step of fusing a shape prior into $\mathbf{v}^m$ is required.

Because each element of $\mathbf{v}$ is, itself, an n-vector coordinate $v_j^m = (x_j^m, y_j^m)$, to talk about the estimation of $\mathbf{v}$ is mathematically ambiguous. Therefore, the estimation of the components $\mathbf{x}, \mathbf{y}$ will be explicitly discussed. Let $\mathbf{v}^e = (\mathbf{x}^e, \mathbf{y}^e)$ be the *true* and unknown discretized deformable model vertices that need to be estimated, considering $\mathbf{v}^e$ to be a



(a) Example active contour

(b) κ vs. s

(c) $\Delta v(s)$ vs. s

Fig. 4.6: (a) A Viterbi boundary. (b) Absolute value of the curvature along the length of the active contour. (c) Arclength of each segment. The circular dots identify six prominent high curvature boundary vertices.

random vector with the components of $\mathbf{v}^e$ satisfying a normally distributed prior model Λ with mean $\boldsymbol{\mu}$:

$$\mathbf{x}^e \sim N(\boldsymbol{\mu}_{\boldsymbol{x}}, \Lambda), \tag{4.19}$$

and

$$\mathbf{y}^e \sim N(\boldsymbol{\mu}_{\boldsymbol{y}}, \Lambda). \tag{4.20}$$

Assuming a linear relationship between the measured and true states, the extracted and resampled measured boundary $\mathbf{v}^m$ representing the measurements of $\mathbf{v}^e$ can be expressed as:

$$\mathbf{x}^m = C_x \mathbf{x}^e + \boldsymbol{\nu}_x \tag{4.21}$$

and

$$\mathbf{y}^m = C_y \mathbf{y}^e + \boldsymbol{\nu}_y, \tag{4.22}$$

where $C_x = C_y = I$ are identity matrices, since each vertex entity is measured, and where $\boldsymbol{\nu} = [\boldsymbol{\nu}_x, \boldsymbol{\nu}_y]$ is the measurement noise, itself having statistics

$$\boldsymbol{\nu}_x, \boldsymbol{\nu}_y, \sim N(0, R). \tag{4.23}$$

A Bayesian estimate [102,103] of $\mathbf{v}$ for each component of $\mathbf{v}$ can be obtained by minimizing the expected error norm, such that

$$\begin{aligned} \hat{\mathbf{x}} &= \boldsymbol{\mu}_x + \left(R^{-1} + \Lambda^{-1}\right)^{-1} R^{-1} \left(\mathbf{x}^m - \boldsymbol{\mu}_x\right) \\ &= \boldsymbol{\mu}_x + K_g \left(\mathbf{x}^m - \boldsymbol{\mu}_x\right), \end{aligned} \tag{4.24}$$

and

$$\begin{aligned} \hat{\mathbf{y}} &= \boldsymbol{\mu}_y + \left(R^{-1} + \Lambda^{-1}\right)^{-1} R^{-1} \left(\mathbf{y}^m - \boldsymbol{\mu}_y\right) \\ &= \boldsymbol{\mu}_y + K_g \left(\mathbf{x}^m - \boldsymbol{\mu}_y\right). \end{aligned} \tag{4.25}$$

The term $K_g = \left(R^{-1} + \Lambda^{-1}\right)^{-1} R^{-1}$ is the Kalman gain, weighting the measurement residual, while the noise covariance matrix R decides the uncertainty of the measured locations. A detail derivation of (4.24) is provided in the appendix A.

The remaining task is to select $\boldsymbol{\mu}$, Λ and R, discussed in the next section.

Algorithm 2 $[Q] = \mathrm{GetQ}(q)$

```
for $j = 1 : q$ do
    $\boldsymbol{\ell} = k - 2 : k + 2$;
    Use circular boundary condition to accommodate boundary vertices;
    $Q(j, \boldsymbol{\ell}) = [-\beta, \alpha + 4\beta, -2\alpha - 6\beta, \alpha + 4\beta, -\beta]$;
end for
```

4.3.1 Bayesian Model Determination

The active contour literature [20,21,23,35] considers the constraints on boundary curvature as the prior model of the object boundary. In practice, such prior models are obtained using an extensive training approach [104]. However an unsupervised, broadly applicable, and non-stationary prior is preferred.

In practice, the continuous integral of some function of $v(s)$ is replaced with a discrete norm over the sampled active contour $\mathbf{v}$:

$$\|A\mathbf{x}\|_2 + \|A\mathbf{y}\|_2 \approx \int_0^1 \left(\alpha\left(s\right) \left| \frac{\partial v}{\partial s} \right|^2 + \beta\left(s\right) \left| \frac{\partial^2 v}{\partial s^2} \right|^2 \right) ds. \tag{4.26}$$

Each row of A asserts some discretized constraint on $\mathbf{x}$ or $\mathbf{y}$, where A is banded (here penta-diagonal, for a second-order constraint). The inverse relationship between prior constraints and prior covariance leads to the constraints A implicitly specifying the prior as

$$\Lambda = \left(A^T A\right)^{-1}. \tag{4.27}$$

The total penalty $\left(A^T A\right)^{-1}$ can be defined using a penta-diagonal matrix Q such that:

$$Q = \left(A^T A\right) = \Lambda^{-1}. \tag{4.28}$$

Although the weighting factors $\alpha(s)$ and $\beta(s)$ are treated as constants, a non-stationary prior is acquired by placing the measured samples non-uniformly on the object boundary, as will be explained in Subsection 4.3.2. The matrix Q containing the prior constraints is computed using Algorithm 2. The deterministic portion of the prior model is the mean, $\boldsymbol{\mu}$. One would normally consider $\boldsymbol{\mu} = 0$ in the absence of any specific deterministic knowledge of the contour shape. However, $\boldsymbol{\mu}$ can be used to create biases in active contour evolution, leading to expansion or contraction forces. The mean $\boldsymbol{\mu} = (\boldsymbol{\mu_x}, \boldsymbol{\mu_y})$ is defined as

$$\boldsymbol{\mu_x} = \boldsymbol{\mu^c_x} + (1 + \boldsymbol{\tau})\, \boldsymbol{\delta_x}, \tag{4.29}$$

and similarly for $\boldsymbol{\mu_y}$, where $\boldsymbol{\mu}^c$ is the center of mass of $\mathbf{v}^m$, and $\boldsymbol{\delta}$ is a circle of points with a radius equal to the average separation of $\mathbf{v}^m$ from $\boldsymbol{\mu}^c$. To avoid bias due to varying active contour point densities, the shape center is normalized with respect to arclength:

$$\boldsymbol{\mu}^c_x = \left(\frac{\sum\limits_{j=1}^{q} x^m_j |x^m_{j+1} - x^m_j|}{\sum\limits_{j=1}^{q} |x^m_{j+1} - x^m_j|} \right). \tag{4.30}$$

The constant $\boldsymbol{\tau}$ in (4.29) is a growth factor on the average radius, creating an expanding force if $\boldsymbol{\tau} > 0$, and a contracting force if $\boldsymbol{\tau} < 0$, motivated from the balloon force active contour [23].

With a prior model $(\boldsymbol{\mu}, \Lambda)$ defined, the key question is the relative weighting of the measurements versus the prior in (4.25), a weighting which is controlled by diagonal covariance R, where

$$R_{jj} = r\left(v^m(s_j)\right). \tag{4.31}$$

For DAC, the variances are treated as a function of their respective image gradients, such that the larger the gradient magnitude $\mathbf{g}$ (2.13), the more certain the measurement and the closer R is to zero. The limiting cases of the measurement variance should be approximated by

$$r(\mathbf{g}) = \begin{cases} r_{\max} & \text{if } \mathbf{g} = 0 \\ \frac{r_{\max}}{2} & \text{if } \mathbf{g} = \mu_{\mathbf{g}} \\ 0 & \text{if } \mathbf{g} = \infty \end{cases}, \tag{4.32}$$

a relationship easily satisfied using

$$r(\mathbf{g}) = r_{\max} \frac{f(\mathbf{g})}{1 + f(\mathbf{g})}, \tag{4.33}$$

where

$$f(\mathbf{g}) = \frac{1}{\sigma_{\mathbf{g}}} \exp\left(-\frac{\mathbf{g} - \mu_{\mathbf{g}}}{\sigma_{\mathbf{g}}} \right). \tag{4.34}$$

The rationale behind f is to scale r on the basis of the average gradient $\mu_{\mathbf{g}}$ and standard deviation of gradients $\sigma_{\mathbf{g}}$ along the observed boundary $\mathbf{v}^m$ in the image. Because noise pixels do not form a continuous boundary, therefore near a noisy region the value of $\mathbf{g}$ is smaller, and that of $\sigma_{\mathbf{g}}$ larger, leading to a larger value of r, implying lower measurement certainty.

The effectiveness of this gradient-sensitive measurement variance is demonstrated in Fig. 4.7. In particular, Fig. 4.7(b) shows the convergence of the active contour by the use of fixed $R = I$, meaning that this fixed R fails to discriminate between the gradients generated by noise and by an object. In contrast, using the image-dependent R from (4.33), the measurements are weakened in noisy regions, giving more weight to the prior, pulling the active contour away from such regions, leading to the excellent convergence shown in Fig. 4.7(c).

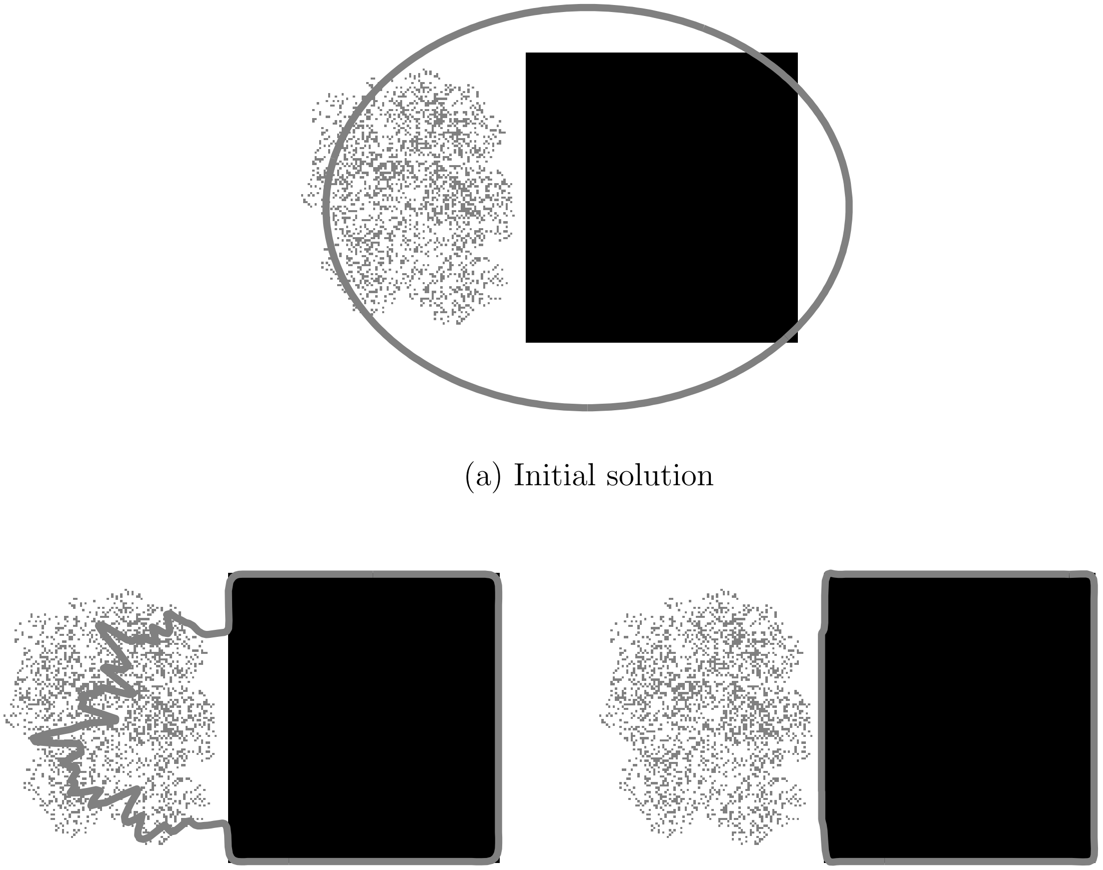
To better understand the role of measurement variance, Fig. 4.8 plots the measurement noise variance (R) for images D, E, H, I, J and M of Figs. 4.10, 4.11 and 4.12; these images are chosen for their varying contrast, non-homogeneity, noise, and background clutter. For uniform contrast foreground and background images (H and J), $\mathbf{r}$ is similar to a step function, because the distribution of image gradient magnitude follows a narrow band (lower $\sigma_{\mathbf{g}}$). For nonuniform contrast and noisy images (D, E and I), $\mathbf{r}$ changes towards a inverted sigmoid function, due to a larger variance in the distribution of image gradient magnitude. For background cluttered images (M), $\mathbf{r}$ is significantly different due to larger values of the mean and variance of image gradient.

4.3.2 Non-Stationary Prior

The prior actually developed in Subsection 4.3.1 has constant $\alpha(s)$ and $\beta(s)$, and is therefore stationary. An argument for a non-stationary prior is motivated in Section 4.2, leading to the rationale for importance sampling. Here, this claim will be evaluated.

Recall from (A.1) the prior

$$\mathbf{x} \sim N(\boldsymbol{\mu}_x, \Lambda) \approx N\left(\boldsymbol{\mu}_x, (A^T A)^{-1}\right) \tag{4.35}$$



(a) Initial solution

(b) Solution for fixed $R = I$

(c) Solution for variable R using (4.33)

Fig. 4.7: Demonstration of the role of measurement noise co-variance (R) in the convergence of active contours in the presence of noise. Panel (a) shows the initial active contour overlapping the square object of interest beside a noise cloud. Panel (b) shows the erroneous solution generated with fixed $R = I$. Panel (c) shows the correct solution using variable R from (4.33). Because the calculated gradient is spatially averaged, gradients near continuous (true) boundaries are stronger than those at discontinuous (noisy) ones.

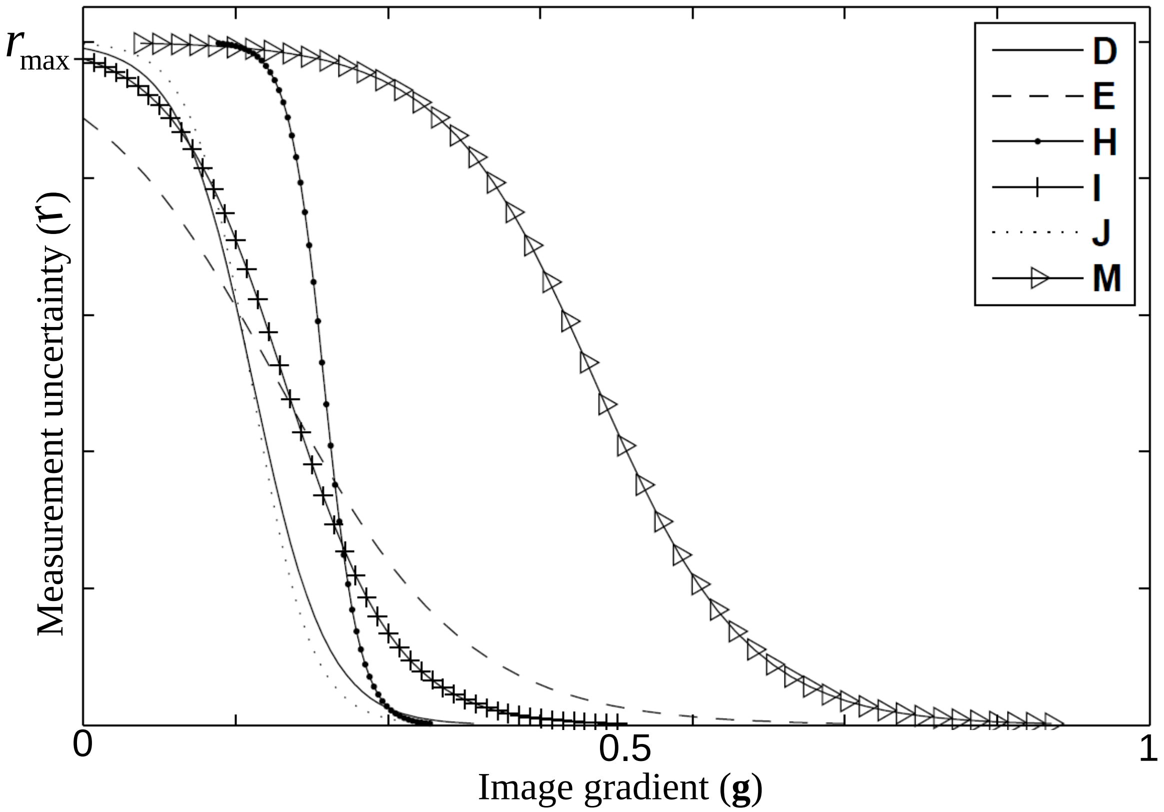


Fig. 4.8: Examples of measurement noise variance $r(\mathbf{g})$ as a function of image gradient for images D, E, H, I, J and M in Figs. 4.10, 4.11 and 4.12.

where A is a linear penalty term,

$$\|A\mathbf{x}\|_2 \approx \int_0^1 \left(\alpha(s) \left| \frac{\partial x}{\partial s} \right|^2 + \beta(s) \left| \frac{\partial^2 x}{\partial s^2} \right|^2 \right) ds \tag{4.36}$$

such that A synthetically approximates the first and higher derivatives on the basis of local differences. Correctly computing a discrete derivative requires taking into account the sampling interval. However because the constraint, represented by A, is *stationary*, and does *not* take the interval into account, therefore the effective result is to induce a non-stationary discretization of the derivative, essentially meaning that $\alpha(s)$ is space-varying, an increasing function of discretization interval.

The resulting phenomenon is illustrated in Fig. 4.9. Suppose we have sinusoidal measurements, but a zero (flat) prior. For fixed values of α and β, as the number of measurement points increases the influence of the prior on the sinusoidal measurements drops, implying a weaker prior. As the samples become sparser, the regularized curve flattens. The net result is an inherently space-varying prior model, penalizing more strongly in areas of low curvature, and less so in areas of higher curvature.

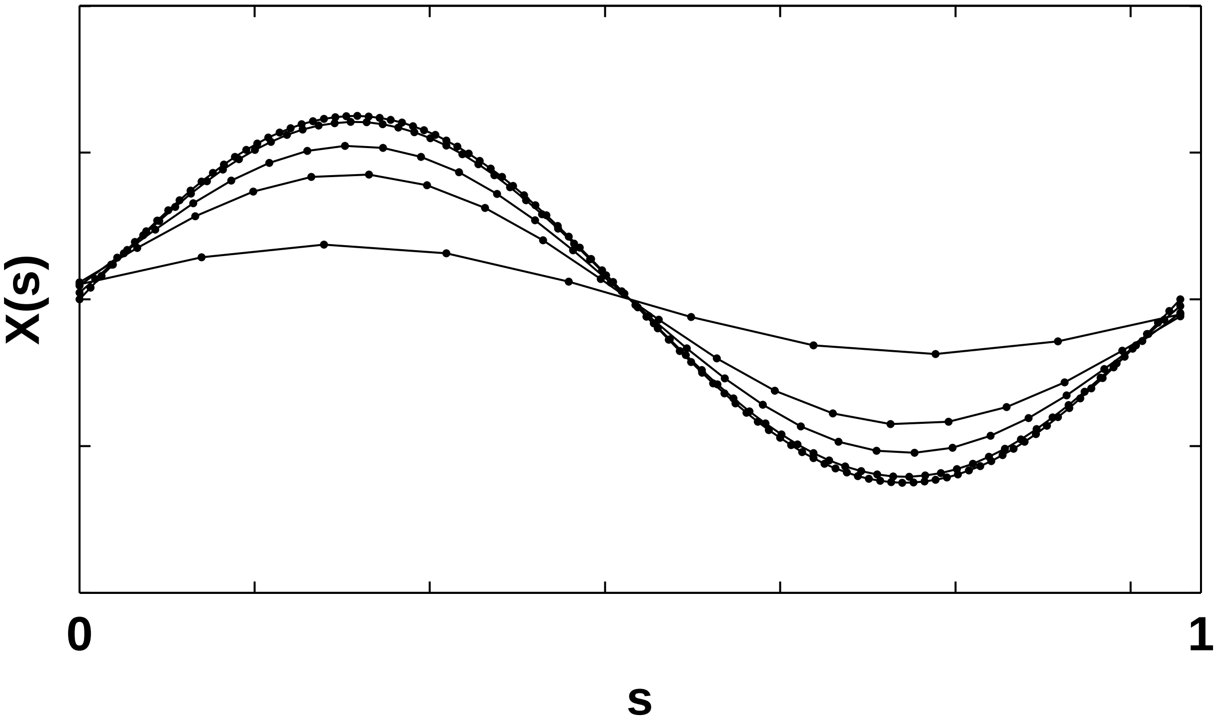


Fig. 4.9: Illustration of regularization as function of the number of samples with constant penalty factor $(\alpha(s), \beta(s))$ and R. As the number of samples decreases the curve inherently becomes more smooth as the prior is asserted more strongly.

The pseudo-code for the complete DAC approach is provided in Algorithm 3.

4.4 Testing and Results

This section describes the experimental dataset, methods compared, experimental setup and demonstrates quantitative and qualitative evaluation of DAC compared to three published methods.

Algorithm 3 $[\mathbf{v}^c]$ = Function DAC($\mathbf{v}_0$)

```
$k = 0$, ASD $= \infty$, $\epsilon = 10^{-4}$, $\mathbf{v}^e_k = \mathbf{v}_0$;
while ASD $\geq \epsilon$ do
    $k = k + 1$, $[\mathbf{v}^m_k]$ = Viterbi($\mathbf{v}^e_{k-1}$);
    Update $\mathbf{v}^m_k$ applying importance sampling on $\mathbf{v}^m_k$ following Section 4.2;
    Estimate $\mathbf{v}^e_k$ using linear Bayesian estimator (4.25) (Section 4.3);
    ASD = GetASD($\mathbf{v}^e_k, \mathbf{v}^e_{k-1}$);
end while
Assign $\mathbf{v}^c = \mathbf{v}^e_k$;
```

4.4.1 Evaluation Dataset and Criterion

Three types of tests are established for evaluating the performance of DAC. Test "A" (Weizmann database [97]) evaluates the average segmentation accuracy of DAC compared to other methods. Test "B" (twenty-one natural and synthetic images) evaluates the quantitative and measured boundary identification accuracy and computational cost. The twenty-one images selected for Test "B" depict various characteristics: 1) complex background and object of interest with weak edges (Images A and D of Fig. 4.10, and R and T of Fig. 4.20), 2) natural and synthetic high curvature images (Images C-H of Figs. 4.10 and 4.11, and N-Q of Fig.4.16), and 3) synthetic and natural, noisy and cluttered images (Figs. 4.12 and Fig.4.16). Test "C" evaluates DAC capabilities mentioned in Section 2.5 through illustrative examples.

4.4.2 Methods Compared

The performance of DAC is compared to the traditional active contour (TS) [20], gradient vector flow snake (GVFS) [21] and active contour without edges (ACWE) [40] using Chan-Vese model. The current DAC implementation is designed for unsupervised segmentation of a single complex object in noisy and cluttered environments. The theory behind DAC is derived from traditional active contour concepts, so TS is considered for a base comparison. GVFS increases the capture range making these approaches less sensitive to initialization.

Non-parametric active contour bases approaches [35–37,40] usually employ more global information in defining object boundaries and can outperform parametric active contours.

Therefore, ACWE [40], a region based approach that is able to capture high curvature regions, is tested. Further, ACWE is designed to work on noisy images by employing a smoothing model to extract the active contour, so ACWE provides a basis on which to evaluate DAC's noise robustness.

For completeness, a sample image is used to compare DAC to level set [37] (LS), intelligent scissors (IS) [9], and greedy snake (GS) [88] methods. These methods have characteristics that do not support extensive test comparisons to DAC. LS is designed for multiple-object segmentation and over-segments the single-object images of this paper. IS requires user interaction, whereas DAC and other methods are unsupervised. GS is tested across all test images, but is unable to segment any image properly.

4.4.3 Experimental Setup

For comparison purposes, published MATLAB code for TS [20], the GVFS [21] and ACWE [40] are downloaded from [105] and [106]. The codes obtained from these websites are written for experimental research purposes without optimization. These codes have been modified using MATLAB vector optimization to allow fair speed comparison with DAC.

The initial location of the active contour is specified manually for all images in Test "B" as shown in Figs. 4.10, 4.11, 4.12, 4.13 and 4.16, while for Test "A" the initial active contour is always chosen as an ellipse. For DAC, TS, and GVFS, the snakes are initialized with $q = 250$ discrete vertices. The suitable range of parameters for TS and GVFS are chosen from [21]. Since ACWE is a region based approach, ACWE is initialized with a closed circular area as shown in Figs. 4.10, 4.11, 4.12, and 4.16, and the optimal parameters are chosen from [40]. For all methods but DAC, σ_s is set using the value provided in the respective papers. For all experiments, DAC uses $r_{max} = 2000$, $\Delta v_{\min} = 0.5$ and $\Delta v_{\max} = 8$, $\alpha = 1, \beta = 0.5$, $\tau = 0$, $\epsilon = 10^{-4}$ and $\sigma_s = 0.5$. For each normal, the number of nodes u is automatically selected so that the normal touched no other nearby normal. The u nodes on each normal are separated by one pixel. Experiments are performed on a 2.4 GHz, 1G RAM, Intel P4 computer.

Table 4.1: Single Segment Average F-measure score (mean ± standard deviation) of DAC, GVFS [21] and TS [20] over 100 images and 94 images suitable for parametric methods. These 100 images are collected from Weizmann database [97].

Algorithm	Average F-measure score	
	Over 100 images	Over 94 images
DAC	0.88±0.060	0.93±0.040
GVFS	0.75±0.120	0.80±0.10
TS	0.72±0.153	0.76±0.13

4.4.4 Test “A”: Segmentation Using Weizmann Database

This thesis evaluates the segmentation performance of DAC on the Weizmann database [97] consisting of 100 images, each of which contains one foreground object, differing from the background on the basis of intensity, texture, or other low level cues. The single object F-measure score [97], a popular segmentation accuracy measure index, is shown in Table 4.1 along with its standard deviation. The F-measure score evaluates the performance of a segmentation algorithm by assessing its ability to correctly segment an object relevant to the ground truth from those that are not relevant to the ground truth.

From the results, the Weizmann database images are found to be not suitable for ACWE [40], because of the variations in intensities of the foreground object. Therefore, this thesis has not reported the F-measure score of ACWE. The methods tested here (DAC, GVFS, and TS) only consider the grey level yet low level features are required to properly segment six of the images. Therefore, the F-measure score [97] for all 100 images and for 94 images suitable for parametric methods is shown in two separate columns of Table 4.1. Over the 100 images in the database, DAC provides higher accuracy relative to the other two tested methods.

4.4.5 Test “B”: Boundary Accuracy and Convergence Speed

To measure the quantitative dissimilarity between converged boundary $\mathbf{v}^c$ and true boundary $\mathbf{v}^t$ the average shortest distance (ASD) is used, defined as the average shortest distance

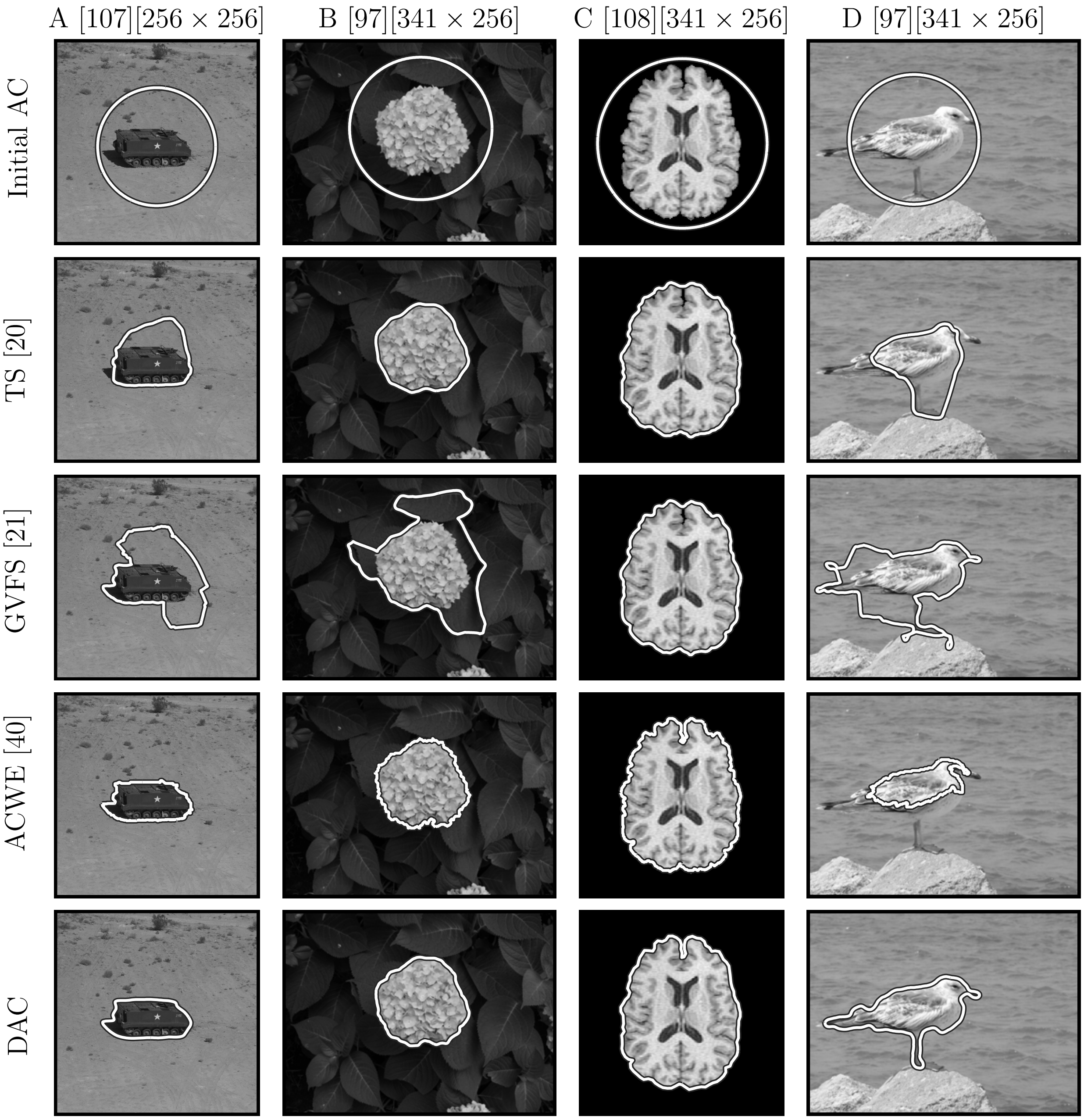


Fig. 4.10: Four active contour methods applied to four different images. Row 1 shows the test images along with the common initial active contour. Rows 2 to 4 show results using three other active contour methods [20, 21, 40]. Last row plots the boundary found by the proposed DAC. In each panel, the white line with black border shows the final contour.

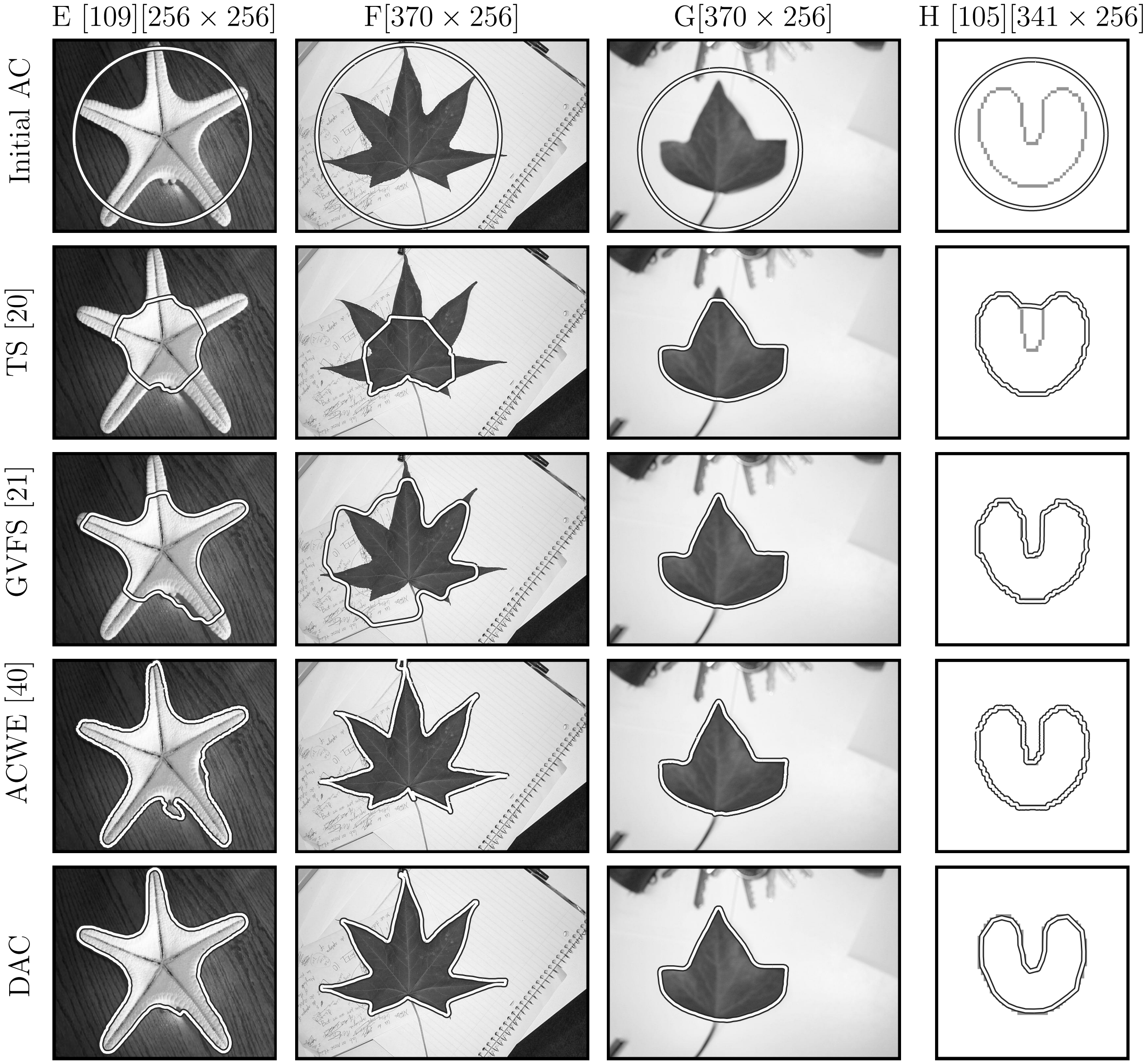


Fig. 4.11: Four active contour methods applied to four different images. Row 1 shows the test images along with the common initial active contour. Rows 2 to 4 show results using three other active contour methods [20, 21, 40]. Last row plots the boundary found by the proposed DAC. In each panel, the white line with black border shows the final contour.

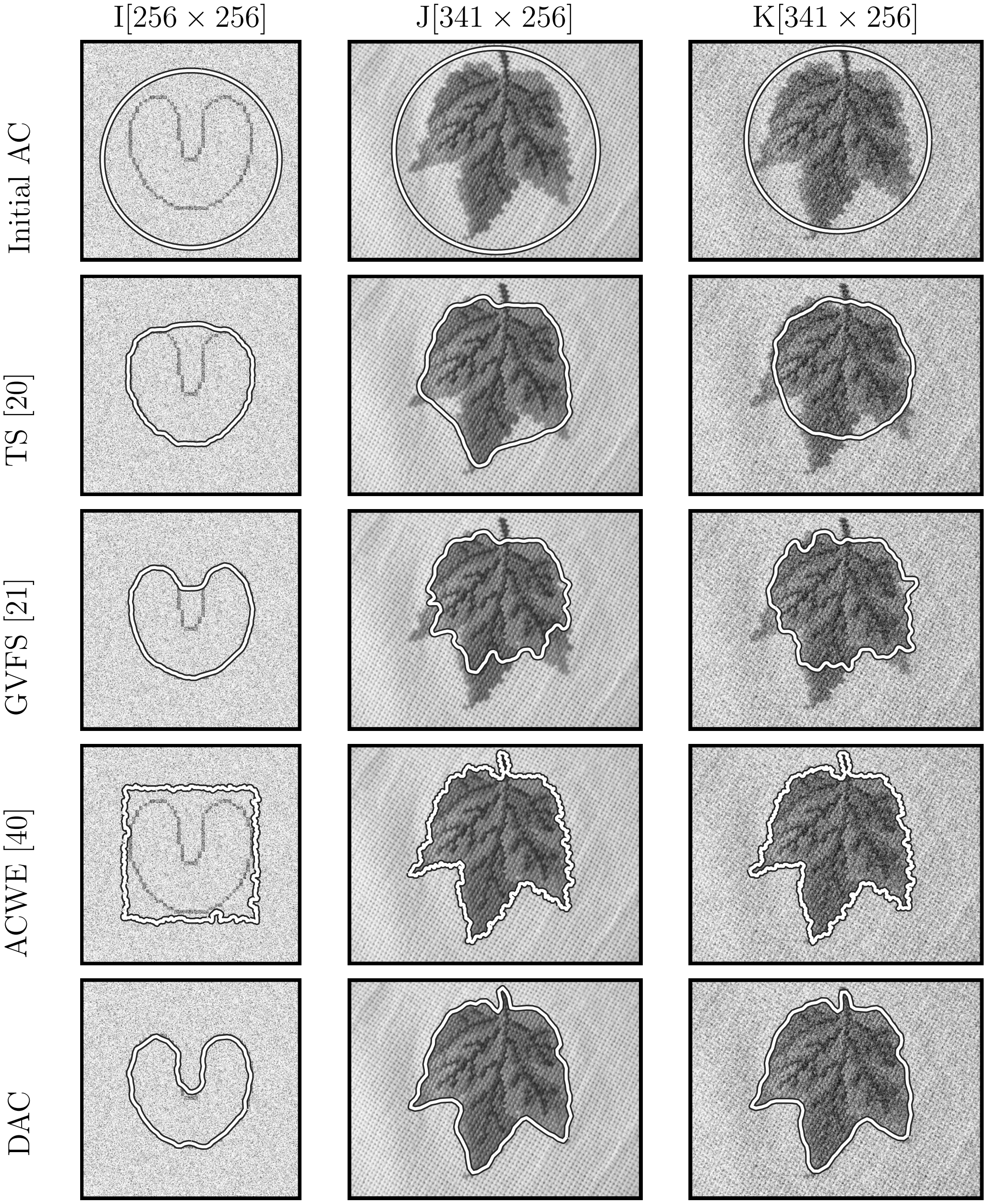


Fig. 4.12: Four active contour methods applied to three different images. Row 1 shows the test images along with the common initial active contour. Rows 2 to 4 show results using three other active contour methods [20, 21, 40]. Last row plots the boundary found by the proposed DAC. In each panel, the white line with black border shows the final contour.

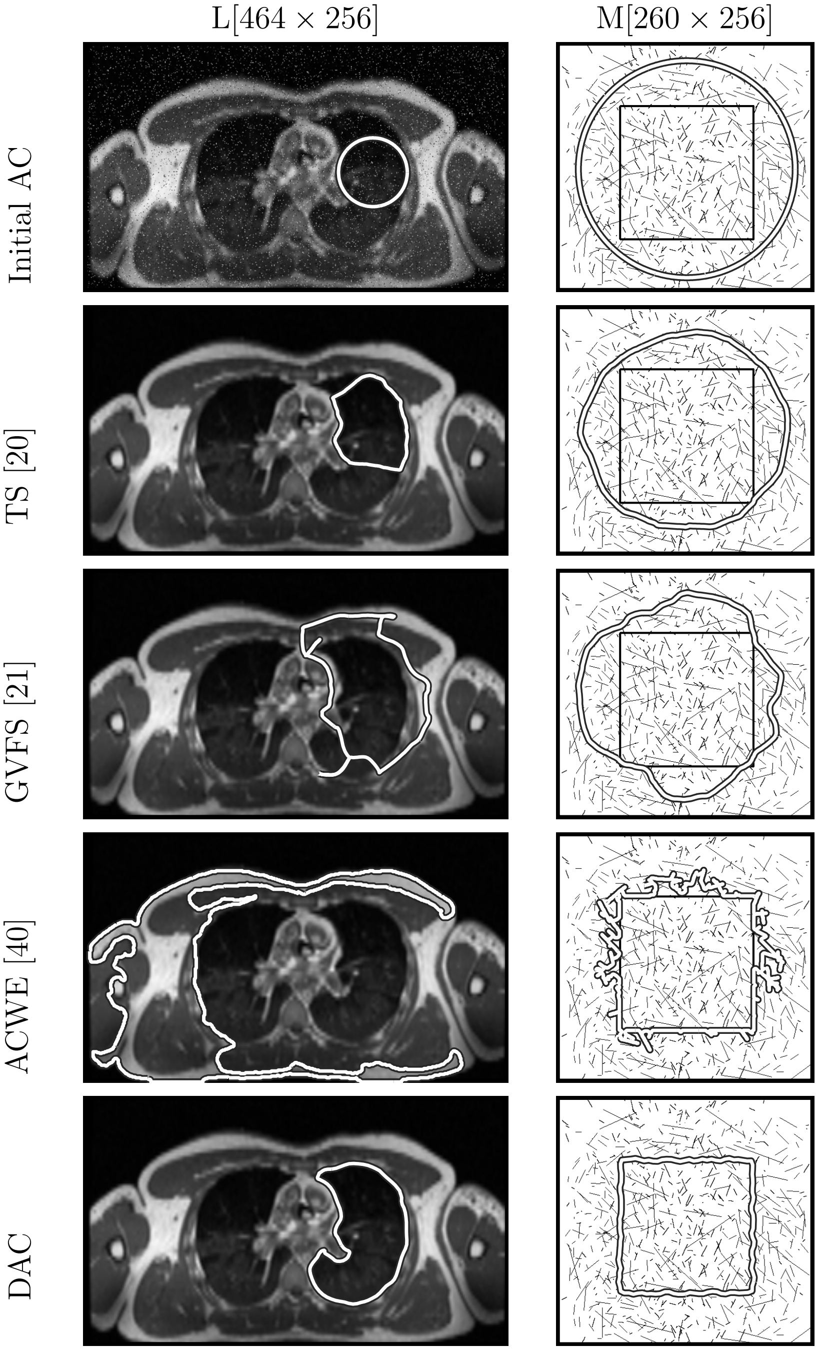


Fig. 4.13: Four active contour methods applied to two different images. Row 1 shows the test images along with the common initial active contour. Rows 2 to 4 show results using three other active contour methods [20, 21, 40]. Last row plots the boundary found by the proposed DAC. In each panel, the white line with black border shows the final contour.

Algorithm 4 $[ASD] = \text{GetASD}(\mathbf{v}^c, \mathbf{v}^t)$

$$ASD = \frac{1}{2q}\sqrt{\sum_{i=1}^{q} \min_{j=1,2\cdots c_n} \left(\|v^t(s_j) - v^c(s_i)\|_2\right)} + \frac{1}{2n}\sqrt{\sum_{j=1}^{n} \min_{i=1,2\cdots q} \left(\|v^t(s_j) - v^c(s_i)\|_2\right)}. \tag{4.37}$$

of the converged boundary from the ground truth and ground truth from the converged boundary and can be computed using Algorithm 4. The number of vertices on the converged boundary (q) and ground truth (c_n) are not necessarily equal.

The qualitative and quantitative results of DAC compared to ACWE [40], GVFS [21] and TS [20] for both natural and synthetic images are presented in Figs. 4.10, 4.11, 4.12 and 4.13, and Table 4.2. Figs. 4.10, 4.11, 4.12 and 4.13 show separate images for the initial contour and the final solution for all four methods. Table 4.2 shows the quantitative values of ASD (4.37) and execution time (ET) for all four methods. Suitable test parameters are chosen from the respective papers of all compared methods for as fair a comparison as possible.

In Figs. 4.10, 4.11 and 4.12, DAC accurately detects the desired object boundary for all images. Because of the non-stationary curvature-dependent prior (Λ) and image statistic dependent measurement uncertainty R, DAC is able to differentiate between true high curvature and the false high curvature (generated due to noise or background clutter) with no parameter tuning. In Table 4.2, DAC is consistently significantly faster and with an accuracy comparable to or significantly better than all compared methods.

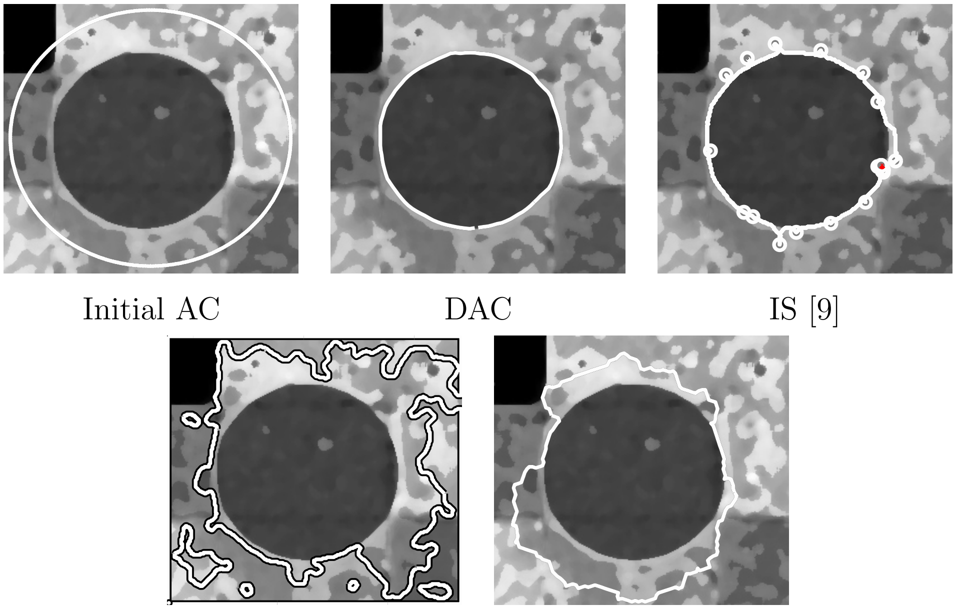
ACWE successfully identified the region of interest for most images of Figs. 4.10, 4.11 and 4.12. As ACWE [40] employs a level set based curve evolution technique to solve the Mumford-Shah model [73] with a piecewise constant approximation, such that the background and foreground can be represented using two constants. Therefore, ACWE model does not use the gradient as the stopping criterion and instead uses intensity homogeneity constraints. In the contrast, ACWE failed for images D, I, L and M because the region of interest can not be described by a single constant (D), ACWE converges to a local minimum generated by noise or clutter (I, M), and because the object of interest is not properly closed (L).

Table 4.2: Average shortest distance (ASD) in pixels and execution time (ET) in seconds of DAC compared to three other methods using thirteen test images in Fig. 4.10, 4.11, 4.12 and 4.20. Bold and italic text indicate best and close to the best performance across all methods for a particular image, respectively.

Average shortest distance (ASD) [pixels]					Execution time [seconds]			
	DAC	ACWE	GVFS	TS	DAC	ACWE	GVFS	TS
(A)	*0.89*	**0.68**	29.17	15.68	**13**	377	208	391
(B)	**0.58**	*0.80*	13.86	*0.63*	**3**	59	109	397
(C)	**1.60**	*1.70*	*2.40*	*2.80*	**7**	70	90	255
(D)	**2.81**	14.26	6.48	9.61	**11**	190	146	351
(E)	**1.40**	*2.70*	9.60	36.9	**5**	159	148	543
(F)	*2.20*	**1.12**	26.62	60.3	**14**	636	357	581
(G)	**0.60**	*0.62*	*0.97*	*2.32*	**6**	612	323	596
(H)	*3.40*	**3.00**	*3.60*	10.6	**5**	134	73	536
(I)	**3.70**	39.9	7.30	13.9	**15**	223	148	683
(J)	**1.60**	*1.69*	8.84	5.81	**10**	281	306	596
(K)	*2.80*	**1.44**	7.96	19.45	**18**	162	189	601
(L)	**3.00**	41.8	9.90	13.55	**16**	171	94	539
(M)	**3.50**	13.44	38.6	43.7	**9**	468	192	603

TS [20] failed to locate the true boundary for most images. TS uses an iterative gradient descent optimization technique (same as for GVFS [21]); as a result, TS is sensitive to local minima and has a slower convergence rate. Furthermore, TS uses local image gradient as the external energy and, as a result, an initial solution far from the true solution may be trapped or take a long time to converge.

To attract an initial active contour from a greater distance, GVFS uses a partial differential equation based diffusion technique to spread out the external energy throughout the image. As a result, GVFS is less sensitive to the initial solution and converges faster compared to TS. The main cause for the failure of GVFS is twofold. First, the diffusion function used for spreading the image potential does not work properly if the nature of the gradient is not simple (Image A, B, D, E F and M). Second, high curvature boundaries



Initial AC

DAC

IS [9]

LS [37]

GS [88]

Fig. 4.14: Performance of DAC, intelligent scissors [9] (LS), level set [37] (LS) and greedy snake [88] (GS) in a cluttered environment.

hinder the GVFS to converge towards the correct solution.

Finally, as a qualitative comparison to other methods, in Fig. 4.14 the segmentation result of DAC is compared to level set [37] (LS), intelligent scissors (IS) [9], and greedy snake (GS) [88]. IS is a user guided technique, therefore the segmentation accuracy of IS depends upon the level of user interactions. Local minima generated due the background clutter prevented the LS to converge towards the right solution. GS is sensitive to initial solution and outliers, as a result it failed to identify the correct boundary of the circular disc image of Fig. 4.14.

4.4.6 Test "C": Evaluation of DAC Capabilities

This section experimentally assesses the DAC in terms of the first five criteria identified in Section 2.5:

1. noise and background clutter,

2. high curvature regions,

3. parameter sensitivity,

4. high curvature regions,

5. initialization sensitivity, and

6. stopping criteria and rate of convergence.

Robustness to Noise and Background Clutter

The performance of DAC in the presence of noise and background clutter is demonstrated in Figs. 4.10, 4.11, 4.12 and 4.13 and Table 4.2 using natural and synthetic images. The DAC accurately identifies the object boundary for all test images. Fig. 4.15 illustrates how the proposed DAC differentiates between true high curvature and high curvature generated due to noise. The first column of Fig. 4.15 shows two images, both having aspects of high-curvature, either noise points (top panel) or the three vertices of a triangle (bottom panel).

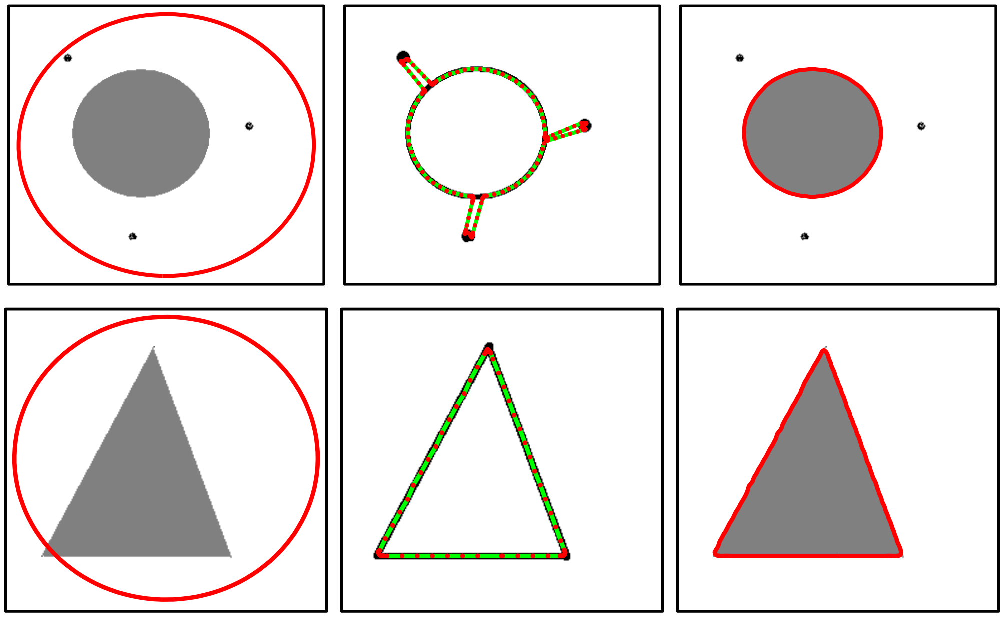
DAC distinguishes between noise and true high-curvature segments in three ways:

1. Because the gradient is computed via convolution, as in (2.13), there is some degree of smoothing, which reduces noise more than structure.

2. Because there is a minimum step size $\Delta v_{min} > 0$, a given noise point will be felt by zero, or at most one, measured vertices, whereas a curvature segment will be felt, and constrained, by multiple measurements.

3. Finally, the computation of the measurement variance r in (4.34) leads to a preference for curves along uniform gradients, rather than curves which encounter varying gradient levels.

The limited number of measurements and greater value of $\sigma_{\mathbf{g}}$ allow DAC to pass through vertices of noise and background clutter (image M of Fig. 4.13).



Initial solution Intermediate solution converged solution

Fig. 4.15: Demonstration of DAC's ability to handle false high curvature (top row) due to noise and true high curvature (bottom row). Each panel shows the active contour, superimposed on the image (or image gradient, in the intermediate case).

High curvature object boundaries

The ability of DAC to capture high curvature regions compared to ACWE [40], GVFS [21] and TS [20] is tested on a set of five synthetic images in which a concavity is made tighter and tighter, a problem of considerable difficulty for most parametric contours. Fig. 4.17 plots the ASD as a function of angle, and results for four of the images are shown in Fig. 4.16. Only the ACWE and DAC perform well on this test set; the GVFS and TS fail to capture the high curvature region as these are both parametric active contours that do not incorporate a non-stationary prior to reduce the penalty near high curvature corners. The DAC's abilities to capture high curvature boundaries is demonstrated via natural images (Test "B") and the tests on the Weizmann dataset (Test "A").

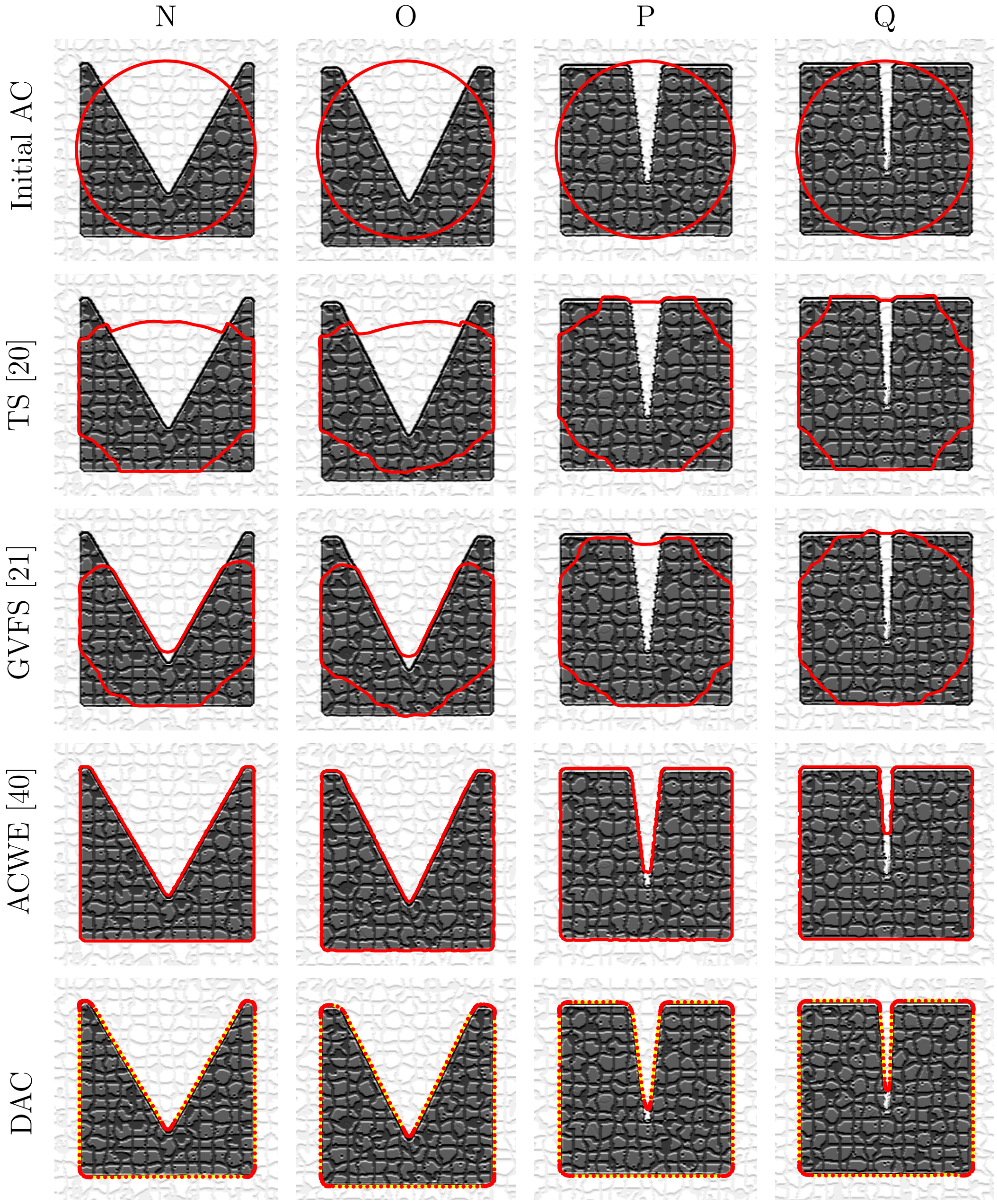


Fig. 4.16: A test of convergence into a concave region, a challenging task for an active contour algorithm. For the DAC, in last row, the dots plot the placement of sample vertices illustrating the curvature-dependent sampling. Of the other three methods, only the ACWE [40] performs similarly. The degree of convergence of the panels in this figure is plotted in Fig. 4.17.

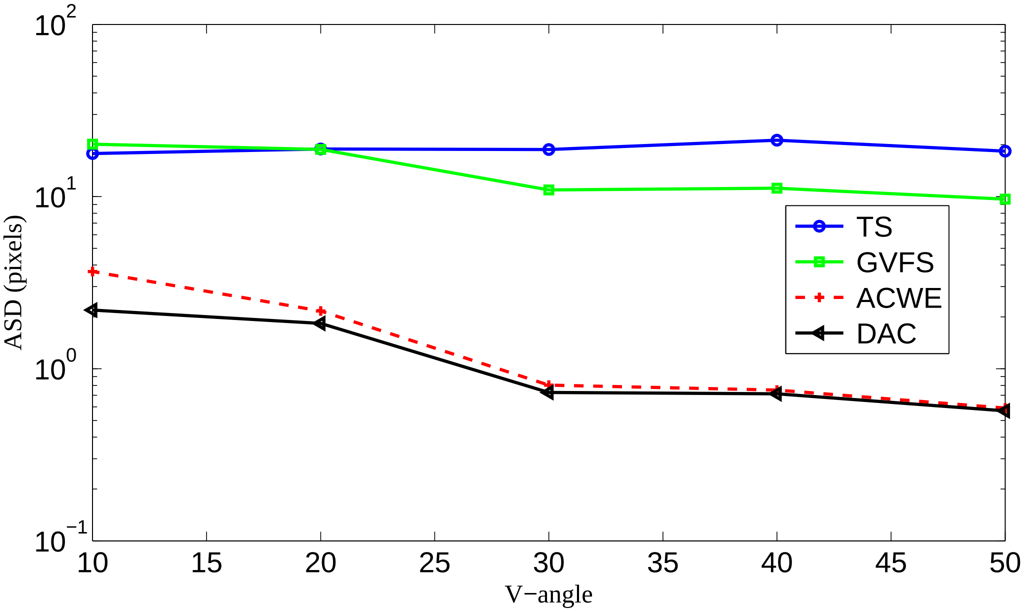
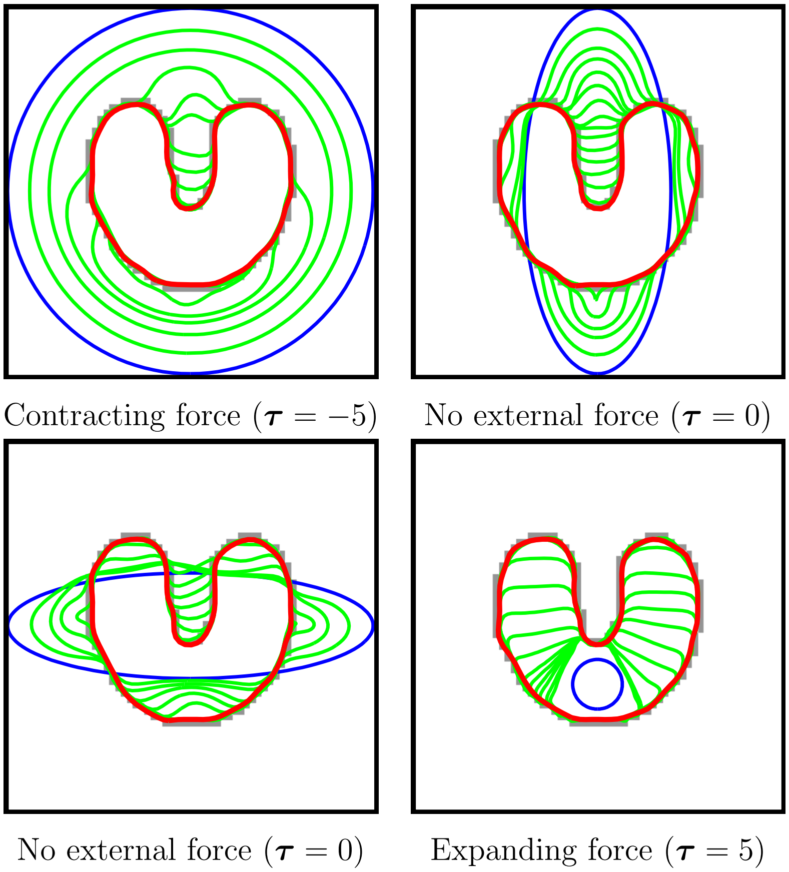


Fig. 4.17: The ASD (Algorithm 4) as a function of V-angle for DAC, ACWE, GVFS and TS from Fig. 4.16. The DAC and ACWE methods are similarly successful, although the DAC outperforms at tighter angles.

Robustness to parameter selection

DAC uses a fixed set of parameters ($\alpha = 1$ and $\beta = 0.5$ in (4.26)) for all of the images in all of the experiments in this chapter. Although most existing methods $[20, 21, 36, 40]$ require the user to manually set a fixed prior and other parameters, a fixed prior is inappropriate for a non-stationary boundary, and image-dependent parameters are inconvenient when working with varied images. The non-stationary prior inferred by the DAC from importance sampling simplifies parameter issues significantly. Essentially, DAC translates the problem of parameter tuning into the problem of seeking a good edge probability by carefully choosing the measurement uncertainty matrix R and non-stationary prior Λ.

As discussed under Test "A", DAC accurately and quickly identified the object boundary for all of these diverse images without a single change to any parameter settings in all cases.



Contracting force ($\boldsymbol{\tau} = -5$)

No external force ($\boldsymbol{\tau} = 0$)

No external force ($\boldsymbol{\tau} = 0$)

Expanding force ($\boldsymbol{\tau} = 5$)

Fig. 4.18: Convergence pattern of DAC for a variety of initial positions given a U-shaped object. For each case, DAC demonstrates insensitivity to initial active contour position by converging to the correct solution in each case.

Robustness to initialization

Fig. 4.18 shows the convergence of four different initializations. Whether the active contour expands or contracts is a function of the external mean $\boldsymbol{\mu}$ (4.30) and (4.29). As shown in Fig. 4.18, the term $\boldsymbol{\tau}$ of (4.29) controls the expansion and contraction of DAC. In contrast, the traditional active contour (TS) requires an initial active contour close to its solution to ensure speed of convergence and accuracy [21]. Although GVFS increases the capture range, the method is not able to find the correct boundary if initialized far from the true boundary. These phenomena can be observed from Figs. 4.10, 4.11, 4.12, and 4.16. ACWE is a region based approach, so it is not sensitive to initial positions.

Stopping criteria and convergence rate

The comparison of computational complexity is ambiguated by the difficulty of assessing convergence in algorithms which converge very slowly. For all examples, as shown in Table 4.2, DAC reached the ASD limit ($\epsilon = 10^{-4}$) quite quickly and so always terminated on the basis of ASD convergence. The value of the ϵ is selected experimentally. For the other compared methods, the slower convergence rate frequently led a simple threshold to lead to early termination. Therefore, the convergence (termination of the active contour evolution) for all images displayed in this chapter are tested manually.

A much more convincing demonstration, independent of stopping criterion, is shown in Fig. 4.19, plotting ASD as a function of computation time. The convergence rate of DAC is fast, roughly two orders of magnitude faster, compared to other methods as noted in Table 4.2. Such an improvement in convergence rate offers potential for more advanced segmentation tasks, such as real-time tracking or three-dimensional problems.

4.4.7 Parametric vs. Non-parametric Methods

Parametric and non-parametric methods of segmentation are fundamentally very different, making any comparison awkward. Aspects of the trade-offs between the two approaches are illustrated in images (R,S,T and U) of Fig. 4.20. Non-parametric active contours inherently identify multiple object (R and S) and can capture high curvature boundaries, since there is no explicitly modeled boundary. Similarly, non-parametric methods are robust to non-uniform and low-contrast boundaries (R). In contrast, non-parametric approaches can not handle discontinuous boundaries (S) and are usually computationally slower than parametric methods.

The DAC make no claims regarding the superiority of DAC, or parametric methods in general, over non-parametric ones. The claim is that for single-object segmentation in the presence of clutter, DAC is able to outperform other parametric and non-parametric methods, both in terms of convergence accuracy and computational complexity. Clearly for multi-object images (R and S) and low-contrast settings (T), non-parametric methods or parametric methods with multiple initialization are the more natural choices.

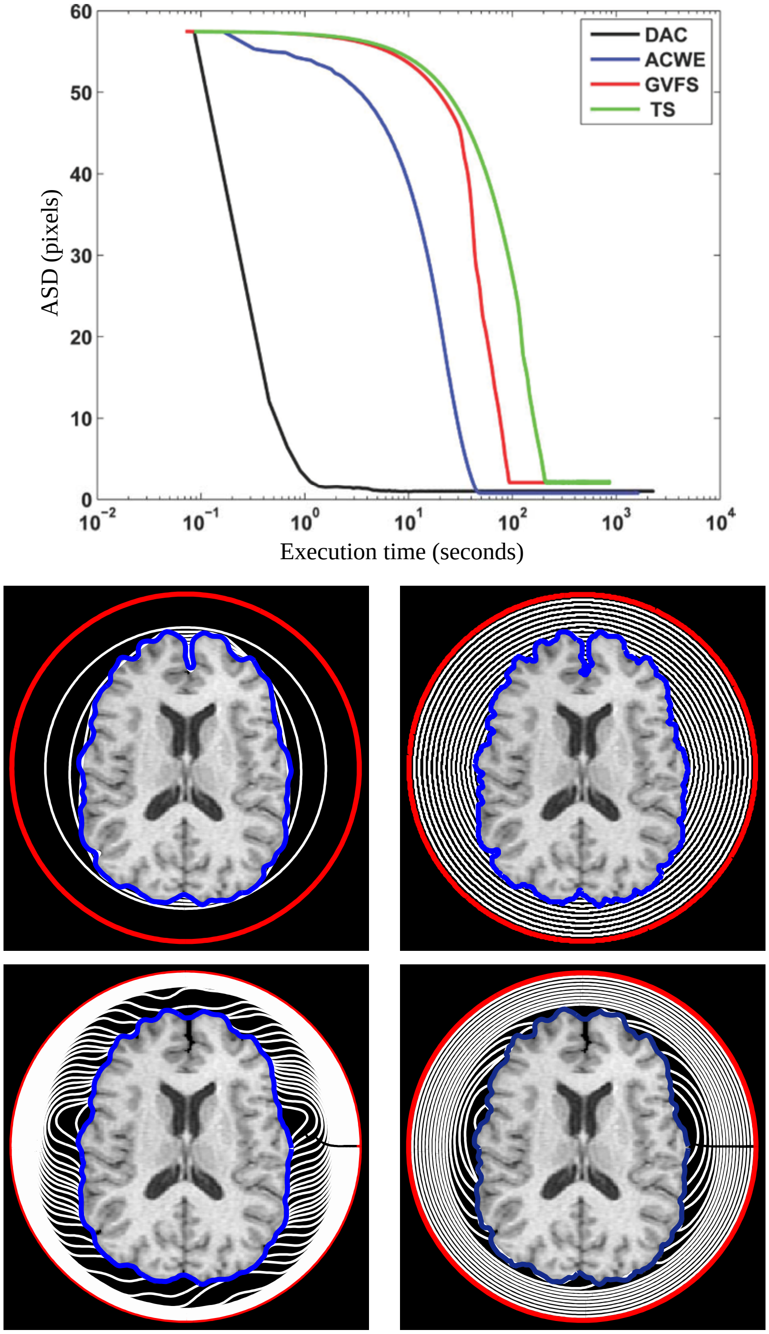


Fig. 4.19: Convergence rate for different methods for brain image. Each method is able to capture the true object boundary but with a different convergence rate. Red, white, and blue lines (second row of Fig. 4.19) are the initial, intermediate and final active contour positions respectively. The ASD from ground truth is shown in first row of Fig. 4.19.

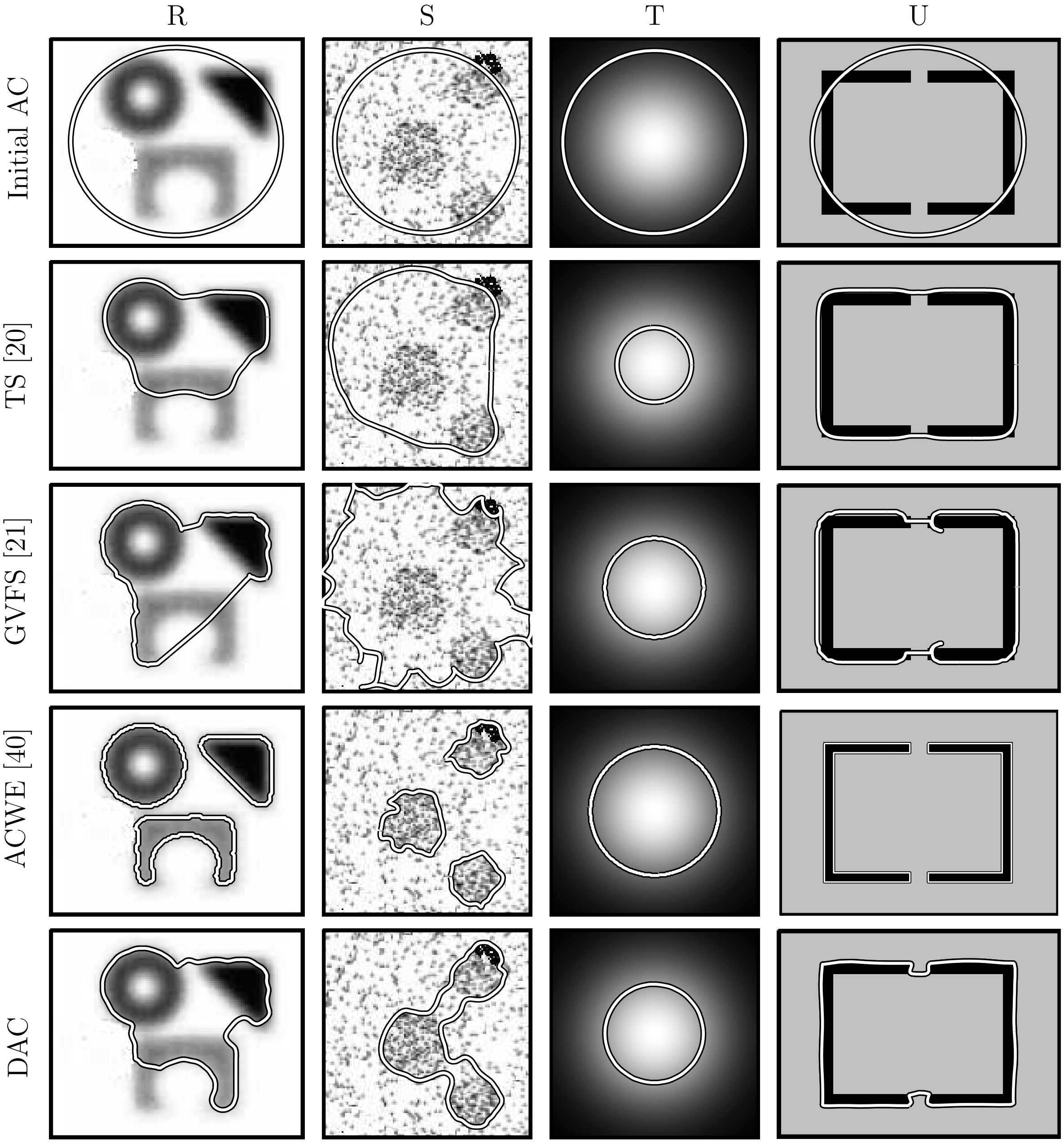


Fig. 4.20: Four active contour methods applied to four different images obtained from [21, 29,97,108,109]. Row 1 shows the test images along with the common initial active contour. Rows 2 to 4 show results using three other active contour methods [20, 21, 40]. Last row plots the boundary found by DAC. In each panel, the white line with black border shows the final contour.

4.5 Summary of DAC

A novel active contour method, the DAC, is designed for rapid and accurate boundary extraction, despite image noise and complex object geometries. Each iteration of the DAC is carried out in three steps: a Viterbi search to find the image gradients, importance resampling to generate a non-stationary prior, and a Bayesian estimator to update the boundary by incorporating prior shape constraints.

Validation of the DAC is demonstrated experimentally on noisy, cluttered natural and synthetic images as well as a standard test database. DAC is demonstrated to be robust to noise, requires no parameter tuning, is able to capture high curvature regions, and is insensitive to initialization and has a much faster convergence rate than compared methods. Of all of the parametric active contours tested for finding the boundary of a single object, DAC is the only one which detected all boundaries accurately. The computational time of DAC is lower relative to other parametric and non-parametric active contours.

One limitation of parametric approaches (such as DAC) is that each active contour is designed to only find a single object. In contrast non-parametric methods are able to identify multiple objects naturally.

Although DAC performed better compared to all examined approaches within the proposed experimental setup, the scalability of DAC to 3D is the next stage of development. Further, it is found that the computational and storage burden of DAC can be further reduced without sacrificing the accuracy by using alternative algorithms for the three steps of the DAC. The next chapter addresses the scalability and computational issues of DAC as a basis for the fast DAC or FDAC.

Chapter 5

Fast Decoupled Active Contour (2D)

Despite the fact that boundary identification techniques have numerous applications in visual tracking [2, 27] and medical image analysis [2, 4, 110], developing a technique to identify the boundary of an object near real-time in the presence of noise and background clutter has been elusive. This chapter introduces the fast decoupled active contour, or FDAC, a fast variation of the decoupled active contour (DAC) for identifying the boundary of a 2D object near real-time. The better computational and scalability performance of the FDAC are compared with the DAC and with two other published methods. Furthermore, the performance gained from FDAC facilitates for effective scaling from 2D to 3D.

5.1 From DAC to FDAC

To increase the convergence speed as well as to avoid the local minima traps typically associated with active contours, DAC is developed and its performance is evaluated in Section 4.4. As described in Chapter 4, DAC separates the prior from the measurements and solves (3.6) by alternating between the measurements and the prior. Essentially, DAC simplifies the active contour energy optimization step by first ignoring the prior and employing the Viterbi algorithm to obtain a sub-optimal solution near the neighborhood of the previous solution that is consistent with the measurements, then generating a non-stationary prior from the curvature of the measured boundary, and finally asserting the non-stationary prior via a Bayesian linear least squares estimator. By using these three steps, DAC precisely identifies the boundaries of high curvature, noisy and cluttered objects with a lower computational burden when compared to state-of-the-art active contour methods. However, the three steps of DAC still suffer from the following computational and scalability issues:

1. As shown in Figs. 5.1 and 5.2, many long normals are required for identifying the boundary of a large image (e.g., 1970 × 1546 pixels). As a result, the computational burden of the Viterbi search ($O(u^2q)$) hinders the possibility of real-time solutions. Furthermore, the causality and spatial order dependency of the Viterbi algorithm (Fig. 4.5) complicates DAC's extension to 3D. Here, the meaning of causality is that the state of a current normal depends only on its previous normals. In 3D, the definitions of "previous" and "next" are ambiguous.

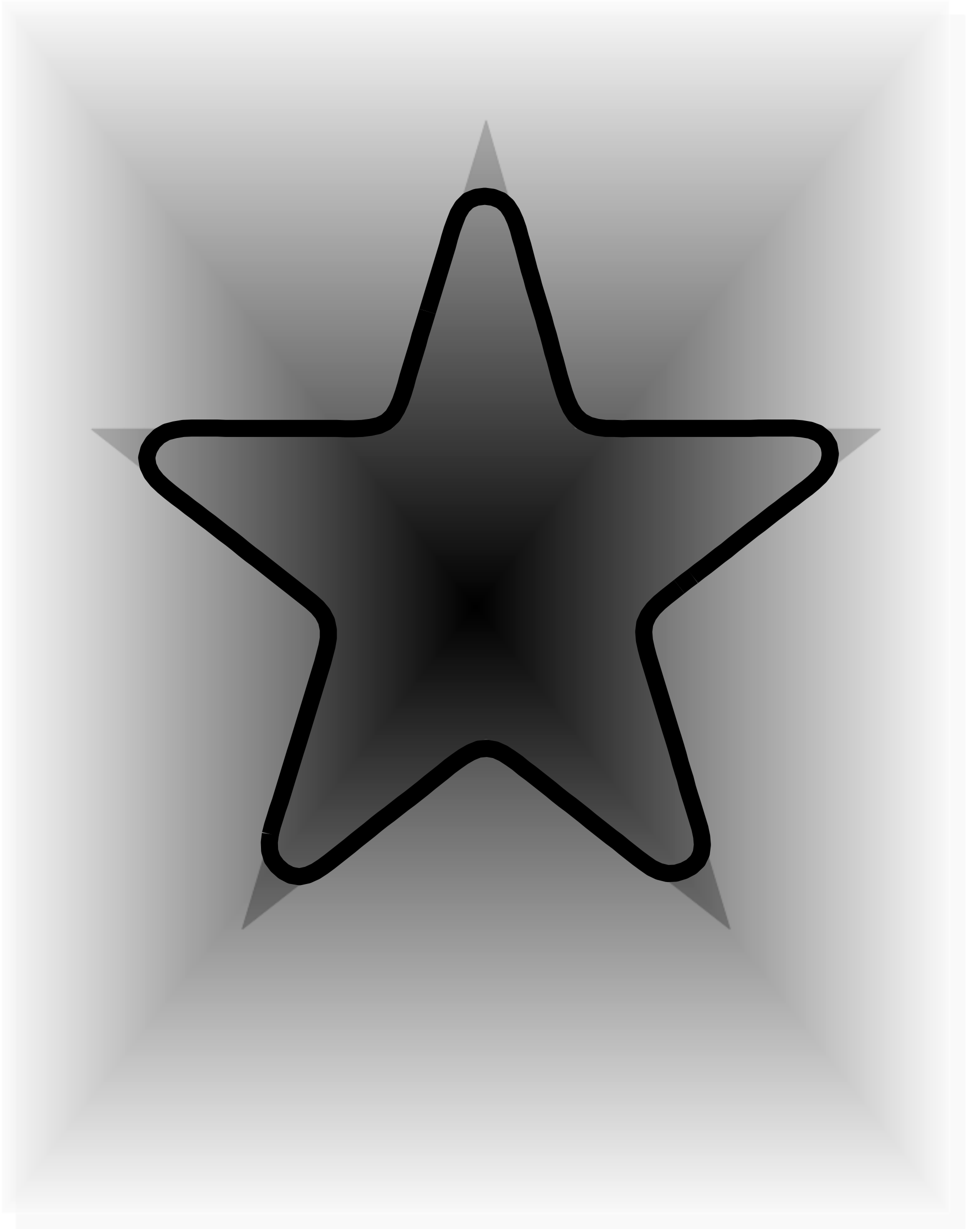
2. The importance sampling step of DAC for generating the non-stationary prior is an ad hoc iterative method, thus typically taking a longer time to generate a curvature based sample distribution.

3. Solving the formulation

$$\hat{\mathbf{x}} = \boldsymbol{\mu}_x + \left(R^{-1} + Q\right)^{-1} R^{-1} \left(\mathbf{x}^m - \boldsymbol{\mu}_x\right) \tag{5.1}$$

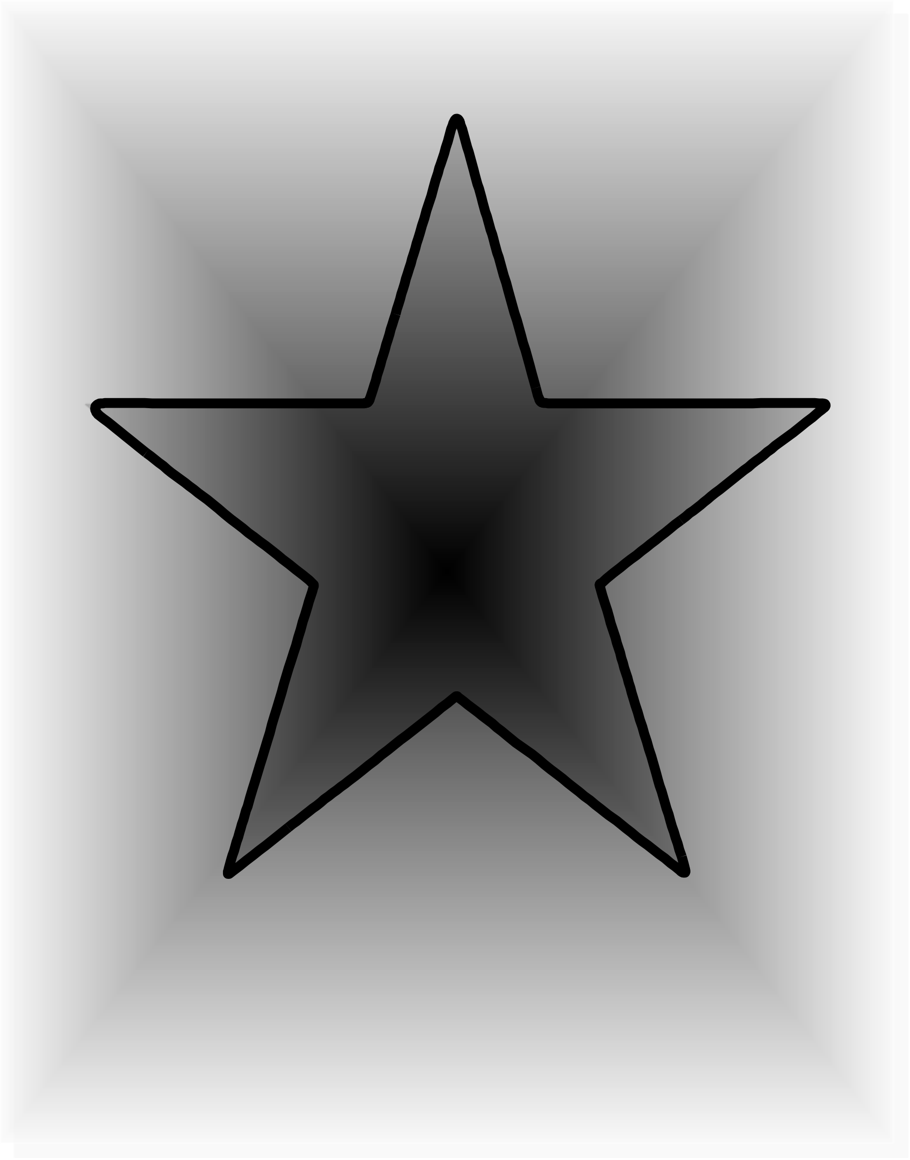
   following the direct matrix inversion approach of DAC for a large number of active contour vertices q (Figs. 5.1 and 5.2) is computationally demanding.

The three computational issues associated with DAC are illustrated using a large image (e.g., 1970 × 1546 pixels) in Figs. 5.1 and 5.2. First, the inferior segmentation accuracy of DAC due to fewer active contour vertices q is demonstrated in Fig. 5.1. Second, the capability of DAC to capture high curvature regions of a large size object using a large number of q is presented in 5.2. Finally, an increase in segmentation accuracy (lower ASD) with the increase in active contour vertices q is shown in Fig. 5.3 using the large image shown in Fig. 5.1. To address DAC's computational demands, a fast decoupled active contour (FDAC) approach is proposed in this chapter. FDAC shares the decoupling idea of DAC; however, to reduce the computational burden of DAC as well as facilitate for efficient extension into 3D, FDAC modifies the measurement (step 1), the resampling (step 2) and the estimation (step 3) steps of DAC using the following three techniques:



Number of active contour vertices $q = 300$

Fig. 5.1: Performance of DAC with fewer number of samples. DAC is not able to capture the high curvature regions with fewer number of active contour vertices q. The identified boundary is shown in black. The size of the image is 1970 × 1546.



Number of active contour vertices $q = 2000$

Fig. 5.2: Performance of DAC with a suitable number of samples. DAC is able to capture high curvature regions by using an adequate number of samples. The identified boundary is shown in black. The size of the image is 1970 × 1546.

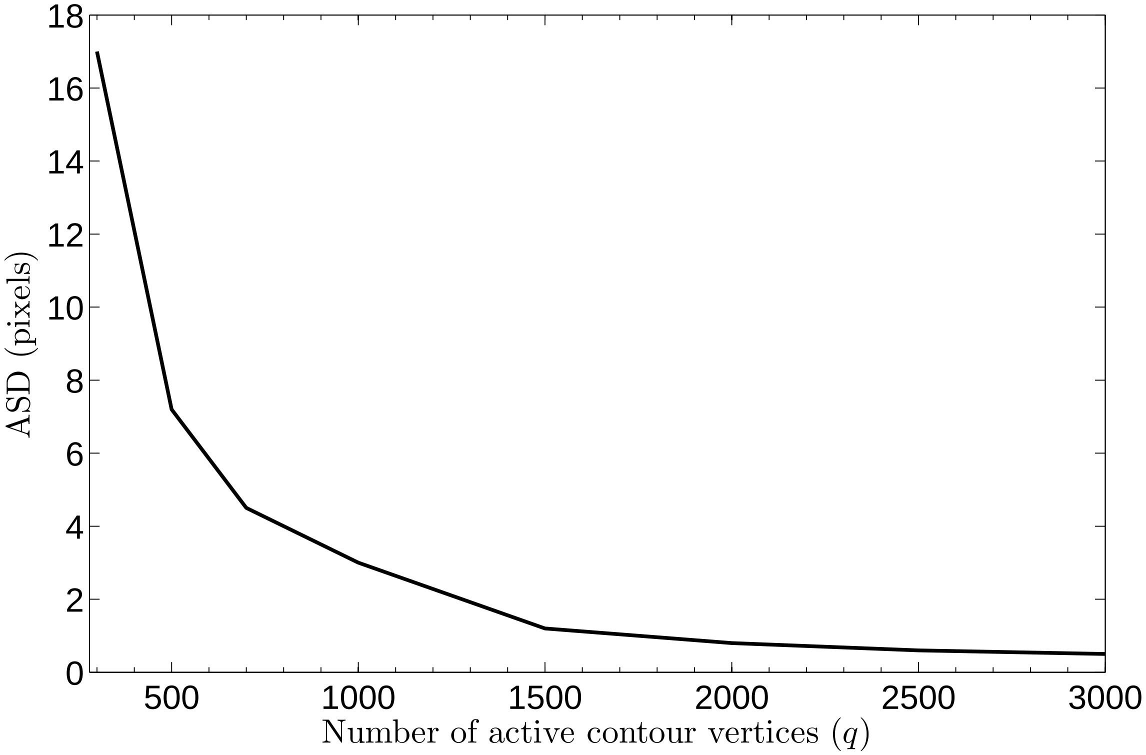


Fig. 5.3: Quantitative segmentation accuracy of DAC with respect to the number of active contour vertices q for the image shown in Fig. 5.1. The segmentation accuracy increases as the number of active contour vertices increases, because a larger number of active contour vertices are necessary to capture the finer details of an object's boundary.

- **Step 1: A fast approach to determine the measured boundary.** An iterative quasi-random search (IQRS) algorithm is used to identify high gradient measured locations as a replacement for the Viterbi search of DAC.

- **Step 2: An efficient technique to resampling.** An uniform sampling scheme based on the integral of the absolute values of the curvature of the measured boundary is employed to generate a curvature dependent measurement as a replacement for the importance sampling step of DAC.

- **Step 3: A Fast approach to statistical data fusion.** A conjugate gradient algorithm is used to perform the Bayesian estimation step efficiently as a replacement for the computationally expensive direct matrix inverse method of DAC.

As with DAC, FDAC is an iterative approach. Therefore, to retain the result of the previous iteration, each iteration of FDAC assumes $\mathbf{v}^p$, the solution from the previous iteration, as the initial guess for the current iteration. As with DAC (Chapter 4), in each iteration of FDAC, the measured boundary $\mathbf{v}^m$ in the vicinity of $\mathbf{v}^p$ is first determined using IQRS, a non-stationary prior is generated by resampling the measured boundary to accommodate high curvature regions, and finally, an estimated solution $\mathbf{v}^e$ is found using the resampled boundary $\mathbf{v}^m$ and the non-stationary prior.

The measurements (Section 5.2), resampling (Section 5.3) and estimation (Section 5.4) steps are explained next.

5.2 Step 1: A Fast Approach to Determine the Measured Boundary

In principle, the measured boundary can be obtained using the Viterbi search method following the descriptions of Section 4.1. However, to improve computational speed as well as to enhance the scalability of DAC, an iterative quasi-random search (IQRS) approach is adopted. The search space or the trellis for the IQRS is defined in an identical way as DAC, where an HMM is used. An exemplar trellis used in IQRS is illustrated in Fig. 5.4. As presented in Section 4.1, given this trellis, in theory, the optimum contour or boundary can be found by solving the joint maximization (4.11)

$$\max_{w_1, w_2, \cdots w_q} P\big(w_1, w_2, \cdots, w_q\big). \tag{5.2}$$

Realizing the difficulty in solving such a joint maximization problem, the Viterbi search based suboptimal algorithm is implemented for DAC in Section 4.1 to find the measured boundary $\mathbf{v}^m$ by maximizing the likelihood distribution

$$\mathbf{v}^m = \arg\max_{\mathbf{v}} \left(P\left(\mathcal{E}_{ext}|\mathbf{v}\right)\right). \tag{5.3}$$

However, the Viterbi algorithm is computationally demanding and scales poorly to higher dimensional problems, as discussed in Chapter 6. Therefore, as an alternative to the

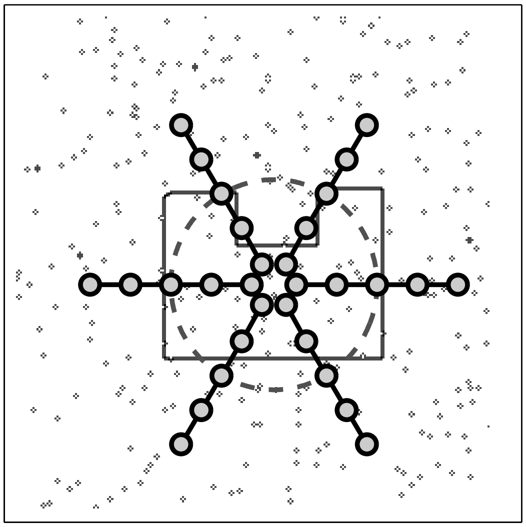


Fig. 5.4: The circular trellis used in the iterative quasi-random search algorithm. The inverse gradient of a noisy u-shaped object, the initial solution (dotted grey circle), as well as the normals (lines with small circular nodes, $u = 4$) are shown to demonstrate the IQRS algorithm. This noisy image is selected to illustrate the necessity of the two steps of the IQRS algorithm (Figs. 5.5 and 5.6).

Viterbi search of DAC, an iterative quasi-random search (IQRS) approach is developed in this section to efficiently determine the measured boundary. The main benefits of IQRS over the Viterbi search are spatial ordering independency, lower computational cost and easer extension to higher dimensions. For simplicity, one iteration of the IQRS algorithm to identify the boundary of a 2D object is explained here.

The IQRS algorithm is simple and consists of two steps: i) initialization and ii) iterative refinement.

Initialization: The initialization step of the IQRS method first computes the probability of all states $w_j \in [-\frac{u}{2}, \cdots, \frac{u}{2}]$, $j \in [1, \cdots, q]$ belonging to a current solution as

$$P(\psi(w_j)|w_j) = \frac{1}{\sigma_{\psi(w_j)}} \exp\left(-\frac{\psi(w_j)}{\sigma_{\psi(w_j)}}\right), \tag{5.4}$$

and then selects states with the highest probability of being on a current solution for the normal j as

$$w_j^m = \underset{w_j \in [-\frac{u}{2}, \cdots, \frac{u}{2}]}{\arg\max} \left(P(\psi(w_j)|w_j) \right), j \in [1, \cdots, q]. \tag{5.5}$$

The notation used in (5.4) is similar to the notation of (4.9). The initial boundary (grey, dotted) obtained using (5.5) is illustrated in Fig. 5.5. This initial boundary is used as a first guess for the iterative refinement step of the IQRS algorithm.

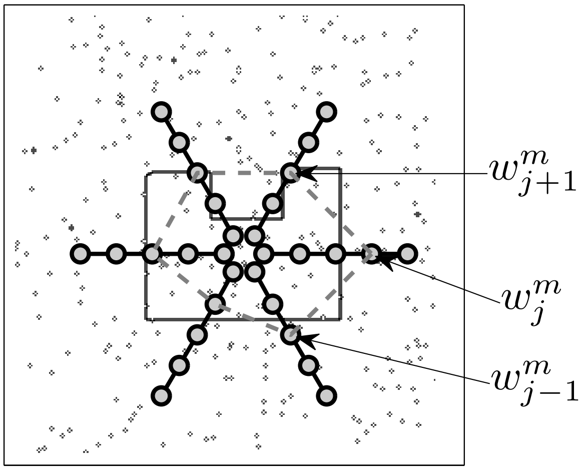


Fig. 5.5: Description of the initialization step of the IQRS method using a noisy u-shape object. The initial boundary obtained using (5.5) is presented in grey (dotted) color. The terms w_{j-1}^m, w_j^m and w_{j+1}^m represent the initial states of the normals $j-1$, j and $j+1$ as obtained by (5.5). These initial states $w_j^m, j \in [1, \cdots, q]$ are used as a first guess for the iterative refinement step (Fig. 5.6) of the IQRS method to update the current state of the normal j.

Iterative refinement: As with DAC's Viterbi search, to avoid unwanted spiky nodes, inter-point constraints between the neighboring nodes of the boundary are enforced in the iterative refining step of the IQRS algorithm. One iteration of the iterative refining step of IQRS follows.

First, a set of q quasi-random numbers **qr** are generated from a Sobol quasi-random sequence [111, 112]. A quasi-random sampling approach is used to allow samples with low discrepancies to be drawn. Second, the inter-point constraint is enforced on a normal qr_j to refine its current state. The second step is repeated for all q normals. As described in Algorithm 5, these two steps are repeated until convergence.

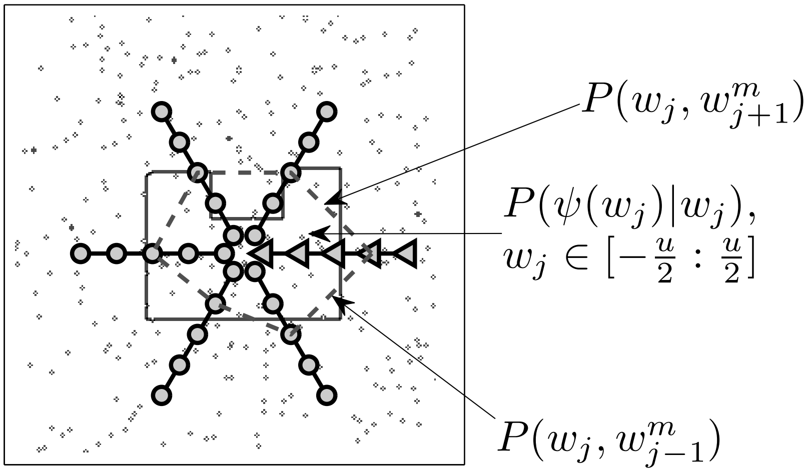


Fig. 5.6: Iterative refinement step of the IQRS algorithm. Inter-point constraints between the states w^m_{j-1} and w_j, and w^m_{j+1} and w_j (pointed by arrows) are computed using 5.8 and 5.8. These inter-point constraints (transition probability) and the observation probability $P(\psi(w_j)|w_j)$ are used to refine the current states ($w_j \in [-\frac{u}{2}, \cdots, \frac{u}{2}]$, here $u = 4$) of normal j (triangular nodes). While refining the normal j, the state of normals $j-1$ (w^m_{j-1}) and w^m_{j+1} are kept fixed. This step is performed for refining all q normals.

The enforcement of these inter-point constraints is demonstrated in Fig. 5.6 using a single normal j (triangular nodes). Where the current state of the normal (with triangular nodes) w^m_j is updated as:

$$w^m_j = \arg\max_{w_j} \left(P\left(\psi(w_j)|w_j, w^m_{j-1}, w^m_{j+1}\right)\right), \tag{5.6}$$

which is equivalent to

$$w^m_j = \arg\max_{w_j} \left(P\left(\psi(w_j)|w_j\right) P\left(w_j, w^m_{j-1}\right) P\left(w_j, w^m_{j+1}\right)\right), \tag{5.7}$$

where

$$P\left(w_j, w^m_{j-1}\right) = \frac{1}{Z_b} \exp\left(-\left|v_{w_j} - v_{w^m_{j-1}}\right|\right), \tag{5.8}$$

and

$$P\left(w_j, w^m_{j+1}\right) = \frac{1}{Z_f} \exp\left(-\left|v_{w_j} - v_{w^m_{j+1}}\right|\right). \tag{5.9}$$

Algorithm 5 $[\mathbf{v}^m]$ = Function IQRS($\mathbf{v}_0$, ϵ)

```
Generate the trellis using $\mathbf{v}_0$ as the initial solution;
$k = 0$, update $\mathbf{v}_0 = \{v_{j,w_j^m}\}, j \in [1, \cdots, q]$ using (5.5);
repeat
  $k = k + 1$;
  Let $\mathbf{qr} = \{\cdots\}_q$ be a sequence of $q$ quasi-random numbers;
  for $k = \mathbf{qr}$ do
    $j = \mathbf{qr}(k)$, update the value of $v_{j,w_j^m}$ using (5.7);
  end for
  Assign $\mathbf{v}_k = \{v_{j,w_j^m}\}, j \in [1, \cdots, q]$;
  ASD = GetASD($\mathbf{v}_k, \mathbf{v}_{k-1}$);
until $ASD \geq \epsilon$
$\mathbf{v}^m = \mathbf{v}^k$;
```

The terms Z_b and Z_f are two normalizing constants used to make $P\left(w_j, w_{j-1}^m\right)$ and $P\left(w_j, w_{j+1}^m\right)$ as two probability distribution functions for normal j. The observation probability $(P(\psi(w_j)|w_j))$ encourages the current state of normal j to lie near high gradient regions. The transition probabilities $(P\left(w_j, w_{j-1}^m\right)$ and $P\left(w_j, w_{j+1}^m\right))$ prevents the current state of the normal j from converging towards a wrong boundary by enforcing spatial constraints between the state w_j^m of normal j, and the states w_{j-1}^m and w_{j+1}^m of normals $j-1$ (previous) and $j+1$ (next).

The pseudo-code of the IQRS for identifying the measured boundary $\mathbf{v}^m$ is provided in Algorithm 5.

Given this measured boundary $\mathbf{v}^m$, a non-stationary prior is essential in the update step to capture the high curvature regions of the measured boundary. The generation of the non-stationary prior (Section 5.3) and the update step (Section 5.4) are provided next.

5.3 Step 2: An Efficient Approach to Resampling

The effectiveness of the non-stationary prior in capturing high curvature regions has been demonstrated in Section 4.2. A curvature guided importance sampling method is employed in Section 4.2 to generate more samples near high curvature regions and fewer samples near smooth regions.

However, the importance sampling technique described in Section 4.2 is an ad hoc and computationally expensive method. Alternatively, curvature guided samples can also be efficiently generated by performing uniform sampling on the absolute value of the curvature weighted arclength of the measured boundary. Performing uniform sampling is typically more straightforward when compared to the importance sampling step of DAC. The resampling process based on the absolute value of the curvature weighted arclength is described as follows.

Let $\boldsymbol{\kappa} = \{\kappa_j\}, j \in [1, \cdots, q]$ be the curvature of the measured boundary $\mathbf{v}^m$. The curvature $\boldsymbol{\kappa}$ of a sample boundary computed using (4.15) is illustrated in Fig. 5.7. Higher curvature near corner regions can be noticed from the bottom panel of Fig. 5.7. Given this curvature profile, the integral of the absolute value of the curvature times the arclength ($kl(s_j)$) over the normalized discrete arclength $s_j \in [0, 1], j \in [1, \cdots, q]$ can be written in the discrete domain as

$$kl\left(s_j\right) = \frac{\sum\limits_{d=1}^{j} |\kappa\left(s_d\right)| s\left(s_d\right)}{\sum\limits_{d=1}^{q} |\kappa\left(s_d\right)| s\left(s_d\right)}, \tag{5.10}$$

with

$$s_j = \frac{\sum\limits_{d=1}^{j} \Delta v_d}{\sum\limits_{d=1}^{q} \Delta v_d} \tag{5.11}$$

being the arclength of an active contour at node j, where

$$\Delta v_j = \left\| v_j - v_{j-1} \right\|. \tag{5.12}$$

The curvature weighted arclength of the measured boundary of Fig. 5.7 is demonstrated in Fig. 5.8.

Given this curvature weighted arclength **kl**, FDAC generates a set of uniform samples, whose sample spacing $\Delta v^u\left(s\right)$ and normalized arclength s_j^u are given by

$$\Delta v_j^u = \frac{kl\left(s_j\right)}{q} \tag{5.13}$$

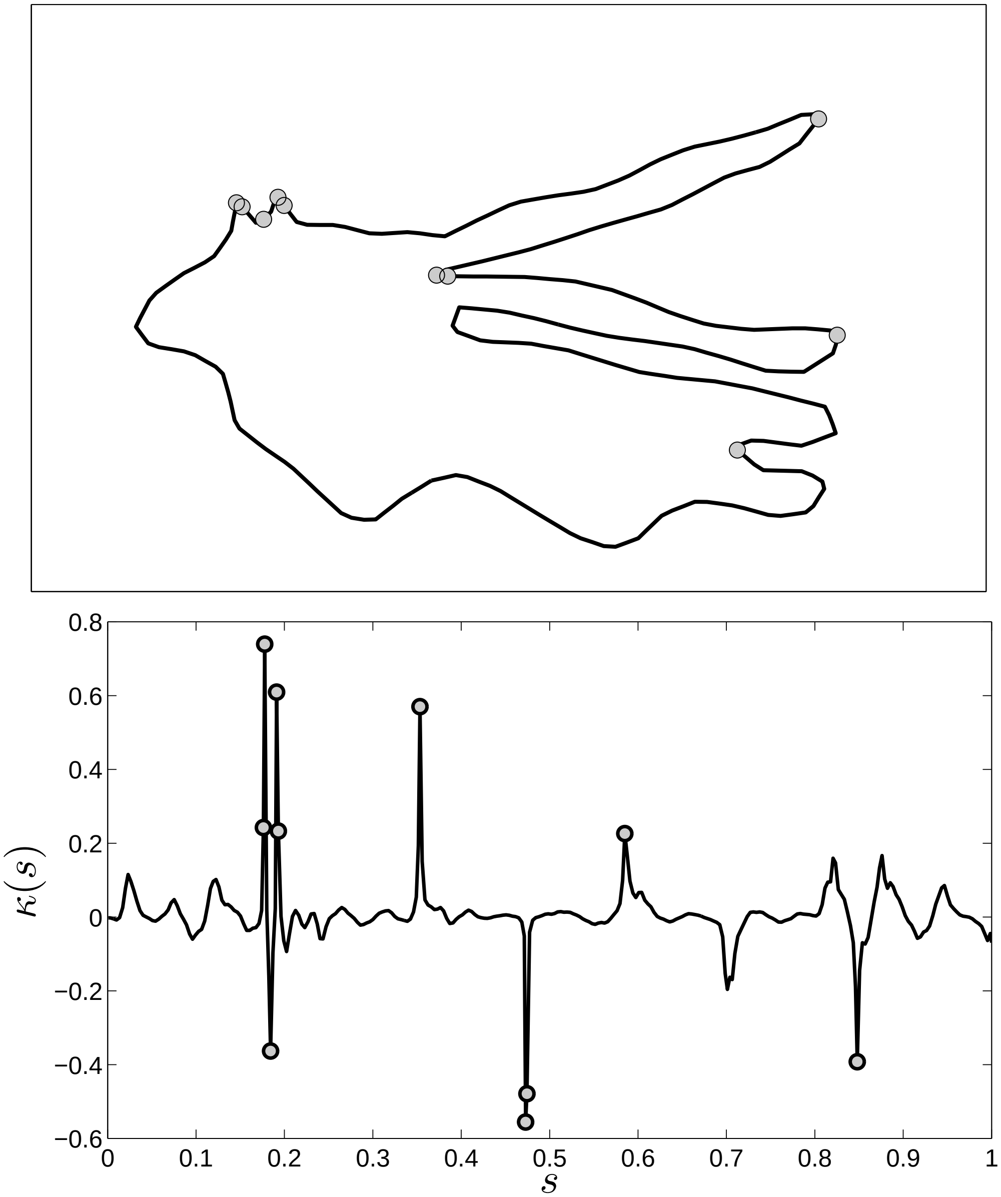


Fig. 5.7: Curvature guided resampling. An exemplar high curvature object boundary (top) and its corresponding curvature $\kappa(s_j)$ (bottom), with the top ten high curvature regions illustrated using small circles. The curvature is computed using (4.15).

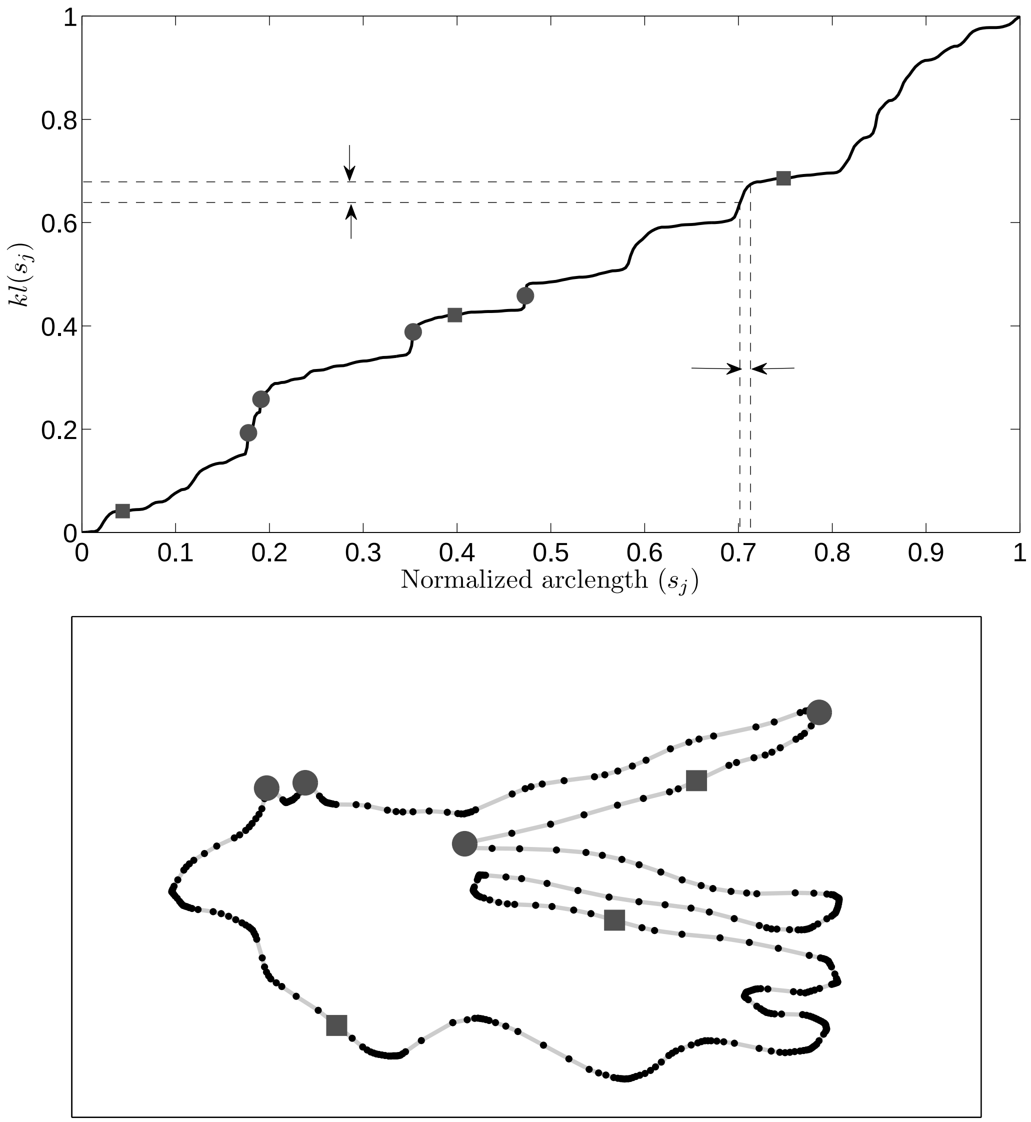


Fig. 5.8: Steps involved in the resampling algorithm. The arclength (s_j) vs. absolute curvature weighted arclength $kl(s_j)$ of the object (top) and the generated samples (bottom) by uniformly sampling the y-axis of the top panel figure are shown in Fig. 5.7, respectively. More samples near high curvature regions can be observed (bottom). The mapping between the top and bottom panels are illustrated using seven key points (grey circles and grey squares).

and

$$s_j^u = \sum_{d=1}^{j} \Delta v_d^u. \tag{5.14}$$

Next, FDAC generates a new set of samples with normalized arclength $\mathbf{s}^n$ from the original sample's normalized arclength $\mathbf{s}$ and the curvature weighted arclength $\mathbf{kl}$ using a spline interpolation function Υ as

$$\mathbf{s}^n = \Upsilon \left(\mathbf{kl}, \mathbf{s}, \mathbf{s}^u \right). \tag{5.15}$$

The generation of these new samples $\mathbf{s}^n$ is demonstrated using a synthetic example in the top panel of Fig. 5.8. Figs. 5.7 and 5.8, which show that near high curvature regions, a wide sampling interval in the curvature weighted arclength space maps to a narrow sampling interval in original Euclidian arclength space, implying a higher sample density near high curvature regions.

Third, FDAC updates the $\mathbf{x}^m$ and $\mathbf{y}^m$ component of the measured boundary $\mathbf{v}^m$ as

$$x^m \left(s_j^n \right) = x^m \left(s_{j-1}^n \right) + \Delta v \left(s_j^n \right) \cos \left(\theta \left(s_j^n \right) \right) \tag{5.16}$$

and

$$y^m \left(s_j^n \right) = y^m \left(s_{j-1}^n \right) + \Delta v \left(s_j^n \right) \sin \left(\theta \left(s_j^n \right) \right), \tag{5.17}$$

where

$$\theta \left(s_j^n \right) = \sum_{p=1}^{j} \kappa \left(s_p^n \right). \tag{5.18}$$

Finally, to avoid generating extremely dense samples at very high curvature regions and extremely sparse samples in smooth regions, FDAC employs an iterative sample insertion and deletion strategy. The iterative sample insertion and deletion strategy enforces the sampling interval to be between Δv_{min} (the minimum sampling interval) and Δv_{max} (the maximum sampling interval). The updated measured boundary $\mathbf{v}^m$ using this curvature based resampling scheme is illustrated in the bottom panel of Fig. 5.8, where a large number of samples can be observed near high curvature regions.

As described in Section 4.3, these non-uniformly distributed measurements on boundary $\mathbf{v}^m$ are used to estimate $\mathbf{v}^e$ and account for a non-stationary prior.

5.4 Step 3: A Fast Approach to Bayesian Estimation

The necessity of enforcing the non-stationary prior to refine the measured boundary using a separate statistical estimation technique has been explained in Section 4.3. This section mainly focuses on developing a computationally efficient method to solve the Bayesian estimation problem. As described in Section 4.3, the linear Bayesian estimation of the $\mathbf{x}$ and $\mathbf{y}$ components of the measured boundary $\mathbf{v}^m$ using the smoothness constraint matrix Q and the measured noise covariance R is given by

$$\hat{\mathbf{x}} = \boldsymbol{\mu}_x + \left(R^{-1} + Q\right)^{-1} R^{-1} \left(\mathbf{x}^m - \boldsymbol{\mu}_x\right), \tag{5.19}$$

and a similar expression holds for $\hat{\mathbf{y}}$. The smoothness constraint matrix Q and the noise covariance matrix R are sparse; hence, $(R^{-1} + Q)$ is also a sparse matrix. Therefore, instead of trying to find the inverse of $(R^{-1} + Q)$, a conjugate gradient based method can be employed to solve (5.19). To utilize the conventional conjugate gradient technique, (5.19) is rearranged as

$$\underbrace{\left(Q + R^{-1}\right)}_{\mathcal{H}} \underbrace{\left(\hat{\mathbf{x}} - \boldsymbol{\mu}_x\right)}_{\mathbf{x}_\theta} = \underbrace{R^{-1}(\mathbf{x}^m - \boldsymbol{\mu}_x)}_{\mathbf{b}}, \tag{5.20}$$

and simplified to

$$\mathcal{H}\mathbf{x}_\theta = \mathbf{b}. \tag{5.21}$$

The algorithm presented in Algorithm 6 is then used to compute Q and the conjugate gradient technique described in Algorithm 7 is employed to solve (5.21) efficiently. Another advanced computational and storage efficient approach to solve large 3D statistical estimation problem is provided in Chapter 6. Finally, the converged boundary $\mathbf{v}^c$ is computed using Algorithm 8.

5.5 Experimental Results

The segmentation accuracy and capabilities of DAC compared to three other state-of-the-art methods has been demonstrated on a wide range of images in Section 4.4. The focus of this chapter is to enhance the computational efficiency and scalability of DAC without sacrificing the segmentation accuracy. Therefore, to avoid unwanted repetition and to

Algorithm 6 $[Q] = \mathrm{GetH}(q)$

```
for $j = 1 : q$ do
    $\boldsymbol{\ell} = k-2 : k+2$, use circular boundary condition to accommodate boundary vertices;
    $Q(j, \boldsymbol{\ell}) = \left[ -\beta, \alpha + 4\beta, -2\alpha - 6\beta + \frac{1}{r_j}, \alpha + 4\beta, -\beta, \right]$, where $r_j$ is computed using (4.33);
end for
```

Algorithm 7 $[\mathbf{x}_\theta] = \text{Function CG}(\mathbf{b}, \mathcal{H})$

```
$k = 0$, $\epsilon = 10^{-4}$, $\mathbf{x}_k = \mathbf{b}$, $\boldsymbol{\varpi}_k = \mathbf{x}_0 - \mathcal{H}\mathbf{x}_k$, $\mathbf{o}_k = \boldsymbol{\varpi}_k$;
while $\mathbf{o}_k \leq \epsilon$ do
    $\text{\ss} = \mathcal{H}\mathbf{o}_k$;
    $\boldsymbol{\lambda}_k = \frac{\boldsymbol{\varpi}_k^T \boldsymbol{\varpi}_k}{\mathbf{o}_k^T \, \text{\ss}}$, $\mathbf{x}_{k+1} = \mathbf{x}_k + \boldsymbol{\lambda}_k \mathbf{o}_k$, $\boldsymbol{\varpi}_{k+1} = \boldsymbol{\varpi}_k - \boldsymbol{\lambda}_k \, \text{\ss}$;
    $\boldsymbol{\rho}_k = \frac{\boldsymbol{\varpi}_{k+1}^T \boldsymbol{\varpi}_{k+1}}{\boldsymbol{\varpi}_k^T \boldsymbol{\varpi}_k}$, $\mathbf{o}_{k+1} = \boldsymbol{\rho}_k \mathbf{o}_k + \boldsymbol{\varpi}_{k+1}$;
end while
$\mathbf{x}_\theta = \mathbf{x}_{k+1}$;
```

ensure FDAC achieves similar segmentation accuracy as DAC, two sets of evaluations are performed. First, the qualitative segmentation performance and quantitative CPU time of FDAC compared to DAC and two other state-of-the-art methods (vector field convolution (VFC) [22] and Poisson inverse gradient (PIG) [54] flow snake) is demonstrated using a set of ten noisy and cluttered images. Second, the FDAC's computational advantage over DAC is evaluated analytically and experimentally using a large synthetic image, as well as two natural images.

5.5.1 Experimental Setup

All the experimental results are obtained using the identical experimental setup as DAC. For comparative analysis, published MATLAB code for Poisson inverse gradient flow snake (PIG) [54] and vector field convolution based active contour (VFC) [22] are downloaded from [113], and the parameters are chosen from the corresponding papers. The initial location of the active contour is always specified using a circle of radius $0.4*(M+N)$ and center at $[0.5M, 0.5N]$ for all methods, with the exception of PIG, whose performance is

Algorithm 8 $[\mathbf{v}^c]$ = Function FDAC($\mathbf{v}_0$)

```
k =0, $\mathbf{v}_k^e = \mathbf{v}_k$, $\epsilon = 10^{-4}$, $ASD = \infty$
while $ASD > \epsilon$ do
    $k = k + 1$, $\mathbf{v}_k^m$ = IQRS($\mathbf{v}_{k-1}^e$, $\epsilon$), Update $\mathbf{v}_k^m$ following Section 5.3
    $\mathcal{H}$ = GetH($\mathbf{x}_k^m$)
    $\mathbf{x}_k^e$ = CG($\mathcal{H}$, $\mathbf{x}_k^m$), $\mathbf{y}_k^e$ = CG($\mathcal{H}$, $\mathbf{y}_k^m$), $\mathbf{v}_k^e = [\mathbf{x}_k^e, \mathbf{x}_k^e]$,
    ASD = GetASD($\mathbf{v}_k^e, \mathbf{v}_{k-1}^e$)
end while
$\mathbf{v}^c = \mathbf{v}_k^e$
```

insensitive to the initial solution. The values M and N represent the number of rows and columns of an image.

5.5.2 Experimental Dataset

Four natural images (starfish (A1), armed personnel control (B1), brain (E1) , and leaf (F1)) and two synthetic images (u-shape (C1) and noise textured v-shape (D1)) images are chosen for this set of experiments. The selected images cover a wide range of characteristics such as contrast non-uniformities, noisiness and contrast in-homogeneities. Furthermore, for validating the noise robustness of FDAC compared to other approaches, another four synthetic images are generated by adding gaussian noise of variances 0.01 and 0.05 and salt and pepper noise of variances 0.01 and 0.03 to the u-shape object. The u-shape object is a standard image used by many parametric active contour methods to evaluate the performance of their methods [21, 22, 54]. Therefore, this thesis has also extensively used the u-shape object for evaluating the performance of FDAC. Finally, for evaluating the computational CPU time of FDAC compared to other approaches, a large (1970 × 1546 pixels) synthetic image with complex boundaries (SCB) has been generated and used.

5.5.3 Segmentation Accuracy

The segmentation results of FDAC compared to PIG, VFC and DAC on four natural, two synthetic, and four noisy images are presented in Figs. 5.9, 5.10 and 5.11. FDAC and DAC are able to successfully identify the boundary of the objects for all images. This is due to the fact that both FDAC and DAC share the same decoupled prior and measurements concept and exploit the image statistics for finding the object boundaries. Furthermore, in the cases of FDAC and DAC, the internal energy not only depends upon the boundary but also the image statistics, while for all other compared methods the internal energy is derived only from the object boundary.

5.5.4 Robustness to Noise

The performance of FDAC in the presence of noise is demonstrated in Fig. 5.11 using a synthetic image with additive Gaussian noise of variances 0.01 and 0.05, and salt and pepper noise of variances 0.01 and 0.03. The FDAC and DAC [55] accurately identified the desired boundary for all test images. VFC [22] claims to be robust to noise and performs better than GVF [21]. However, VFC is typically not effective in discriminating between the true gradient and the gradient due to the high noise or background clutter. Hence, VFC found incorrect boundaries for images with higher noise variances (Fig. 5.11) as well as images with background clutter (Fig. 5.11). PIG uses an additional automatic technique to determine the initial solution near the desired solution. As such, PIG is less sensitive to initialization and local minima. Furthermore, PIG implicitly smoothes the noise during the course of computing the external energy field. Hence, PIG successfully identified the boundary of most test images.

5.5.5 Convergence Test

FDAC uses the same decoupling concept as DAC, but employs more efficient measurements, sampling and estimation schemes to reduce the computational and storage burden. To compare the computational burden of one iteration of FDAC with one iteration of DAC, an initial boundary with q number of vertices is chosen. FDAC replaces the Viterbi search of DAC with the IQRS method and uses the conjugate gradient method to solve the linear Bayesian estimation problem. Therefore, the computational cost of IQRS of FDAC with the Viterbi search of DAC is first compared.

As IQRS of FDAC and the Viterbi search of DAC use the same trellis, the complexity of IQRS and the Viterbi search for a trellis consisting of q normals with each normal containing u nodes are $O(i_{IQRS}uq)$ and $O(u^2q)$, respectively. As described in Section 5.2,

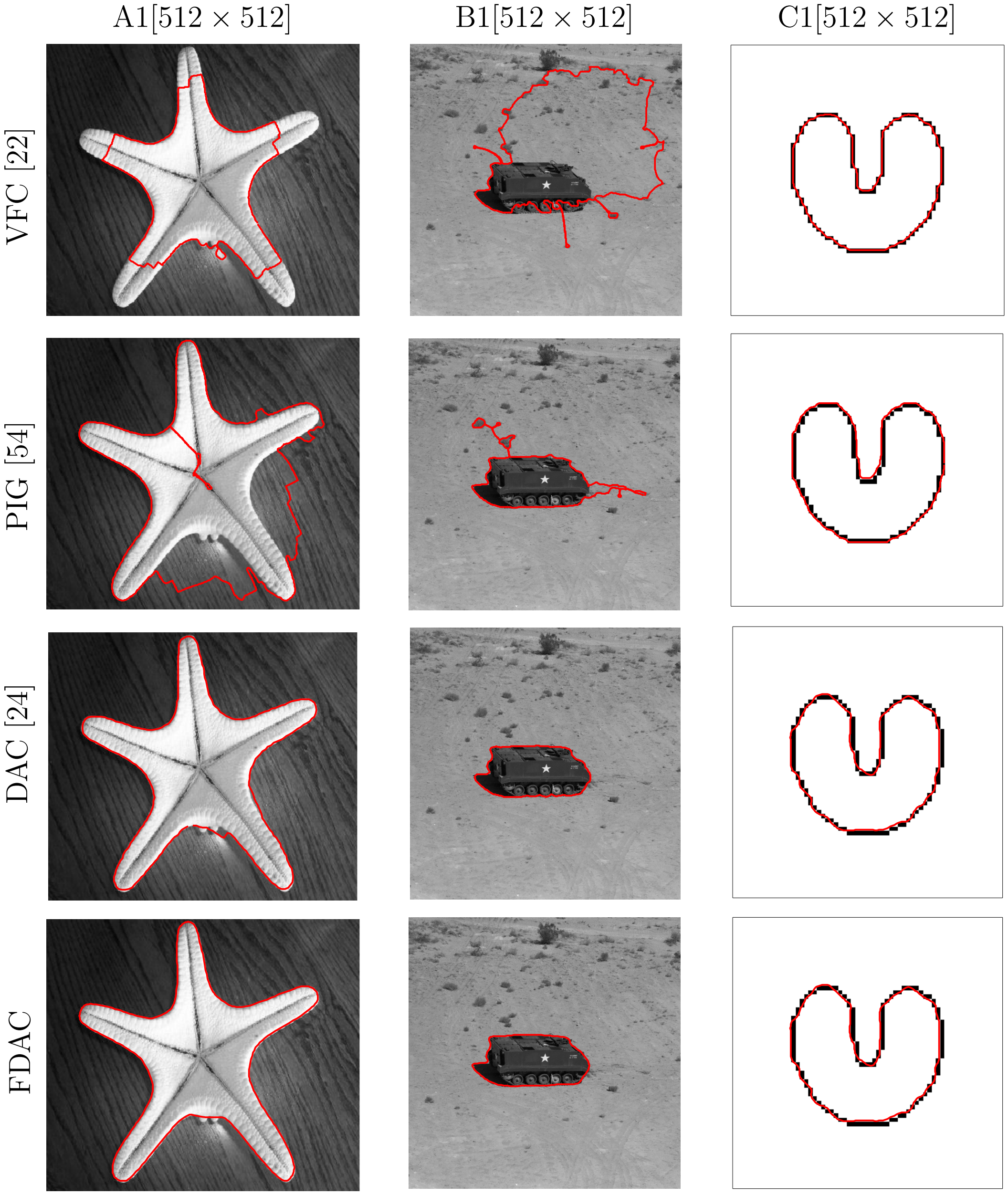


Fig. 5.9: The segmentation performance of FDAC compared to PIG [54], VFC [22] and DAC [24] for two natural images and one synthetic image. Rows present methods and columns present images. For all images and methods, with the exception of PIG which is insensitive to the initial solution, the active contour is initialized using a circle of radius $0.4 * (M + N)$ and center at $[0.5M, 0.5N]$. FDAC and DAC successfully identified the true boundary for all images.

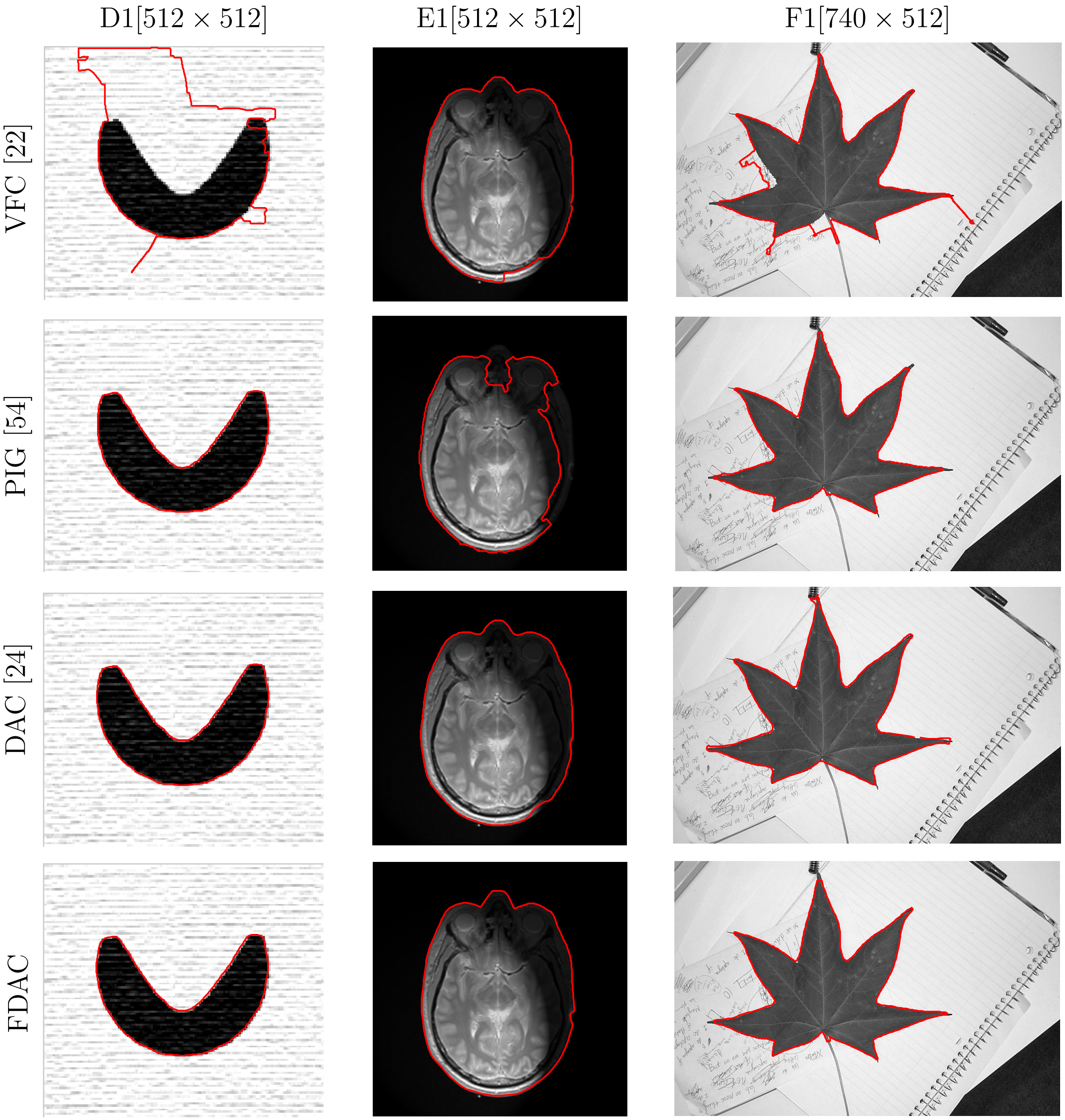


Fig. 5.10: The segmentation performance of FDAC compared to PIG [54], VFC [22] and DAC [24] for two natural and one synthetic images. Rows present methods and columns present images. For all images and methods, with the exception of PIG, the active contour is initialized using a circle of radius $0.4 * (M + N)$ and center at $[0.5M, 0.5N]$. FDAC and DAC successfully identified the true boundary for all images.

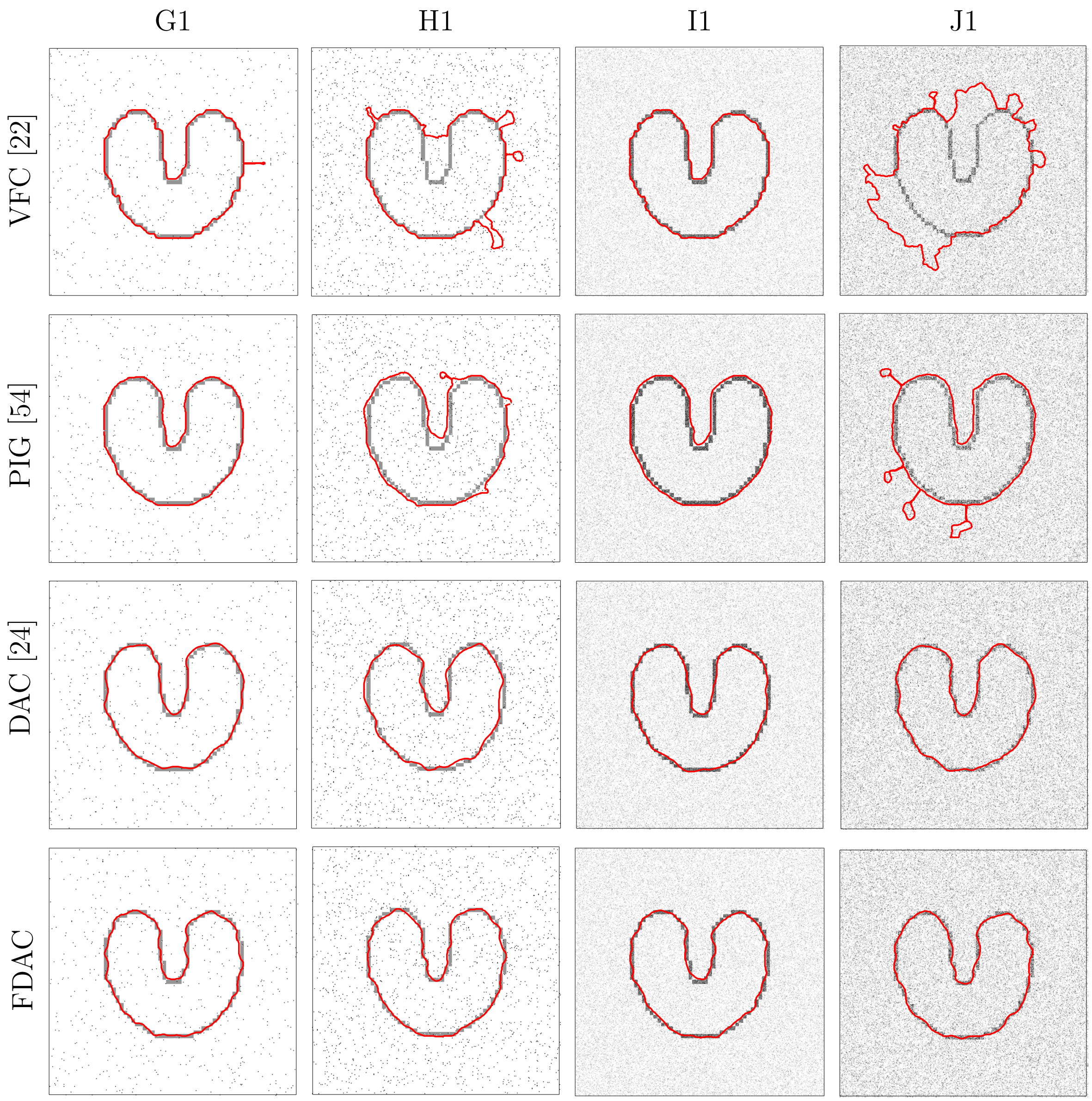
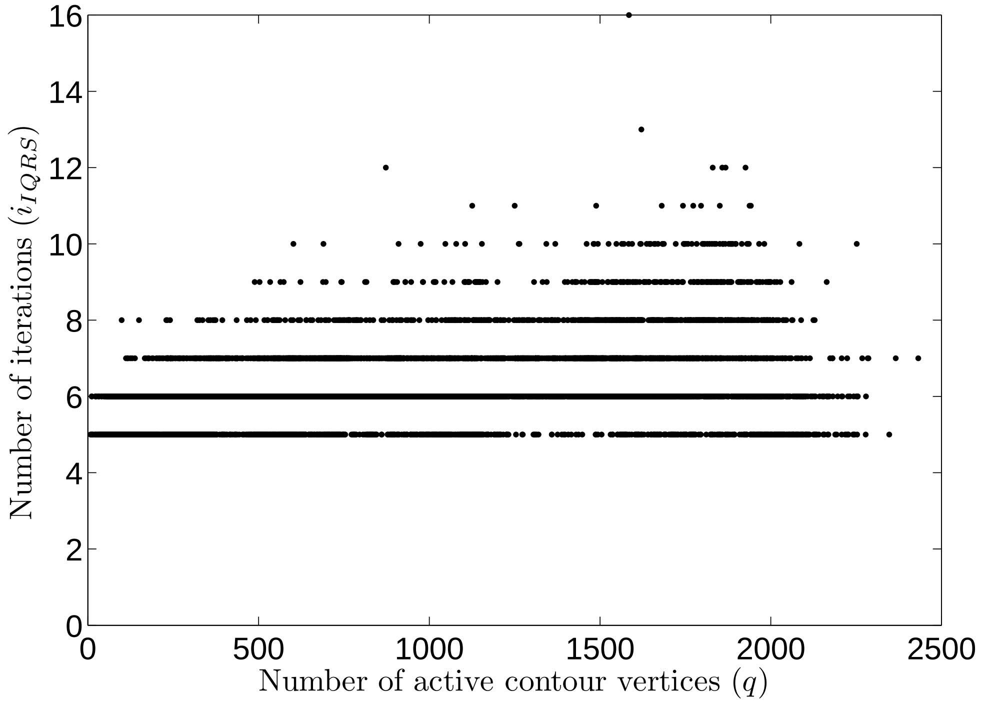


Fig. 5.11: The noise robustness of FDAC compared to PIG [54], VFC [22] and DAC [24] on a synthetic u-shape object. The four noisy images are generated by adding gaussian noise of variances 0.01 and 0.1 and salt and pepper noise of variances 0.01 and 0.03. Rows and columns present images and methods. FDAC and DAC successfully identified the true boundary for all noisy images. PIG successfully identified the boundary of all images except J1.

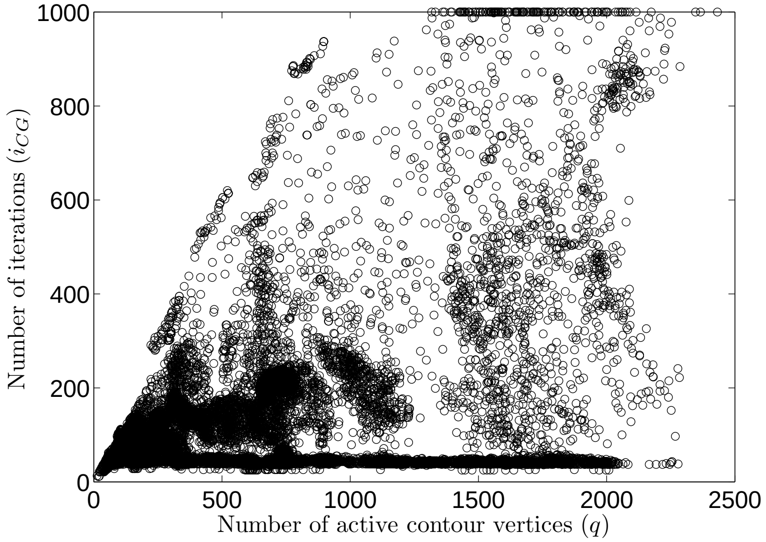


Convergence of IQRS

Fig. 5.12: The convergence plot of IQRS. The plot is obtained using different values of minimum and maximum spacing over three images describing various features such as noise, background clutter and different geometry. The number of iterations required for the IQRS to converge to a desired solution as a function of the number of vertices q. The minimum and maximum number of iterations for IQRS are fixed at 5 and 15.

IQRS is an iterative method and finding an analytical value of the number of iterations i_{IQRS} for IQRS to converge to a desired solution is an extremely difficult task. Therefore, an extensive testing of IQRS's convergence rate with several initial solutions having number of vertices ranging from $q = 50$ to $q = 2500$ are conducted on several images consisting of noise, background clutter and different geometry. The scatter plot of the number of iterations i_{IQRS} of IQRS with respects to number of vertices q is shown in Fig. 5.13. On average, the number of iterations i_{IQRS} required for the IQRS to converge is found to be 7.

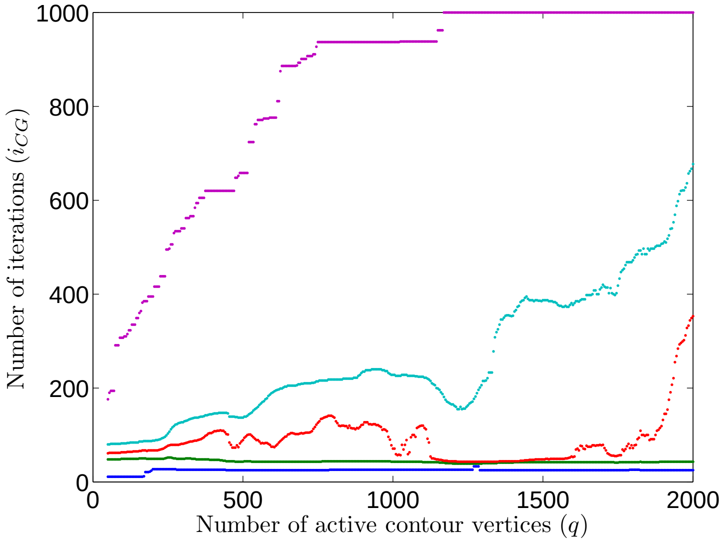
Next, FDAC uses a conjugate gradient (CG) method to update the measured bound-



Convergence of CG

Fig. 5.13: The convergence test of the conjugate gradient method using similar experimental setup as Fig. 5.12. The number of iterations necessary for the CG method to converge to a desired solution depend upon several factors, such as the number of active contour vertices, measurements uncertainties, and the prior. A statistical analysis of this scatter plot is presented in Fig. 5.14.

ary, as opposed to the direct matrix inversion method used in DAC. The complexity of the proposed CG method is $O(i_{CG}q)$, where i_{CG} is the number of iterations the CG method takes to converge. The convergence rate of the conjugate gradient method used in FDAC depends on several factors, such as the dimension of the matrix, regularization constant, geometry of the active contour, and noise variance. Therefore, a scatter plot and a statistical quantitative analysis plot of the convergence of the CG method are shown in Figs. 5.13 and 5.14. An increasing trend in CG iterations can be observed for larger number of vertices. As shown in Fig. 5.13, for all experiments, the maximum number of iterations for the conjugate gradient method is fixed at 1000. To interpret the scatter plot of Fig. 5.13 is a difficult task. Therefore, a statistical analysis of Fig. 5.13 is presented in Fig. 5.14. The five lines from top to bottom of Fig. 5.14 demonstrate the maximum to minimum number of iterations required for the CG method to converge to a desired solution as a function of the number of active contour vertices under different scenarios. The average number of



Convergence of CG

Fig. 5.14: Analysis of the computational complexity of the CG method. A statistical interpretation of the scatter plot of Fig. 5.13 is demonstrated in this panel. Where the five lines from top to bottom demonstrate the maximum to minimum number of iterations required for the CG to converge to a desired solution as a function of the number of active contour vertices under different scenarios.

iterations i_{CG} required for the CG method to converge to a desired solution is found to be 70. Therefore, the complexity of the CG method ($O(i_{CG}q)$) is less when compared to the direct matrix inversion method of DAC.

The quantitative convergence time of FDAC compared to PIG, VFC, and DAC are presented in Table 5.1. FDAC and DAC separates the measurements from the prior and finds the object boundary iteratively. Due to the decoupled nature of the measurements and prior, the FDAC and DAC took fewer iterations compared to VFC and PIG. On average, FDAC is two orders of magnitude faster compared to other three methods. PIG employs iso-surface approach on the poisson inverse gradient of the image to find the initial active contour. As a result, PIG finds an initial estimate of the active contour very near to the optimal active contour. Therefore, PIG took less number of iteration compared to VFC. Although FDAC and DAC shares the same theory of decoupling the measurements from

Table 5.1: Comparison table showing Execution Time (ET) in second of FDAC compared to PIG [54], VFC [22], and DAC. Text in bold letter indicates best performance among their peers for a particular image.

	FDAC	PIG	VFC	DAC
(A1)	**7**	253	513	12
(B1)	**21**	78	816	47
(C1)	**4**	32	114	7
(D1)	**6**	34	670	15
(E1)	**13**	25	163	33
(F1)	**31**	355	982	71

the prior, FDAC uses: i) a faster converging IQRS approach for measurements compared to the Viterbi search approach of DAC, ii) faster uniform sampling on the integral of the absolute value of the curvature compared to importance sampling approach of DAC, and iii) an efficient implementation of conjugate gradient approach for update step compared to direct matrix inversion approach of DAC. The average computational (over six images of Figs. 5.9 and Figs. 5.10) gain of each step of FDAC compared to DAC is illustrated in Fig. 5.15 using a bar plot. For large size images with large number of active contour vertices, FDAC is faster than DAC and this phenomenon is presented in Fig. 5.16.

The convergence times of FDAC, DAC, VFC and PIG as a function of number of active contour vertices are tested on a big synthetic star shape image (Fig. 5.1) of size 1970×1546 pixels. From Fig. 5.16, it can be observed that the convergence time of FDAC is significantly lower compared to DAC for large number of active contour vertices. VFC and PIG both use identical external energies as well as an efficient energy minimization technique. However, unlike VFC, PIG also uses an additional automatic algorithm to determine the initial solution that is near to the desired boundary. Therefore, PIG is insensitive to initialization and takes fewer iterations to converge when compared to VFC. The efficient energy minimization technique and the effective initialization of PIG allows it to achieve better convergence rates when compared to DAC for higher number of active contour vertices.

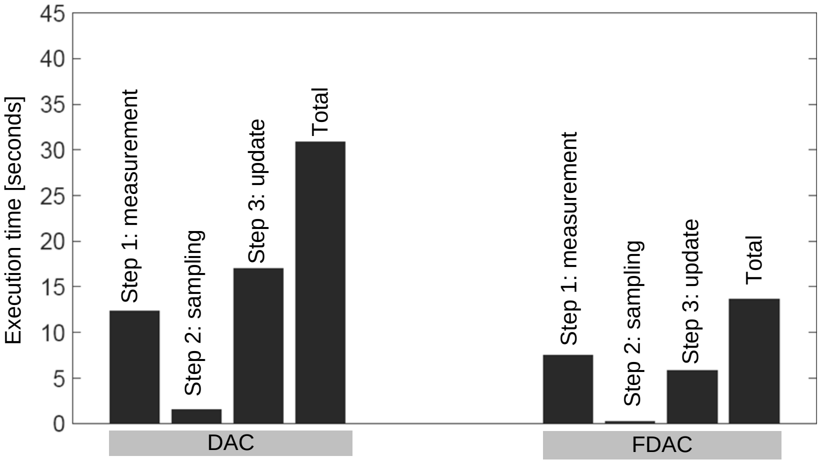


Fig. 5.15: A bar plot of the execution time (seconds) of the three steps of DAC and FDAC over six images (Figs. 5.9 and 5.10). Each step of FDAC is faster compare to each step of DAC. On average FDAC is 2-3 times faster than DAC.

Furthermore, the importance of the number of active contour vertices for capturing high curvature region is demonstrated in Figs. 5.1, 5.2 and 5.3. Fig. 5.3 reports the increase in segmentation accuracy with respect to number of active contour vertices using three different images and different values of minimum and maximum spacing. The Figs. 5.1 and 5.2 show the segmentation result with two different sets of active contour vertices.

5.6 Summary of FDAC

A fast decoupled active contour (FDAC) method for accurate and fast image segmentation is presented in this chapter. FDAC shares the same theory as decoupled active contour (DAC), but employs faster computationally efficient iterative quasi-random search for measurements, a uniform sampling on the integral of absolute value of the curvature of the measured boundary for generating non-stationary prior, and an efficient conjugate gradient based estimation technique to gain computational advantage over DAC.

The segmentation accuracy and convergence time of FDAC compared to other three state-of-the-art active contour methods are reported on several natural and synthetic im-

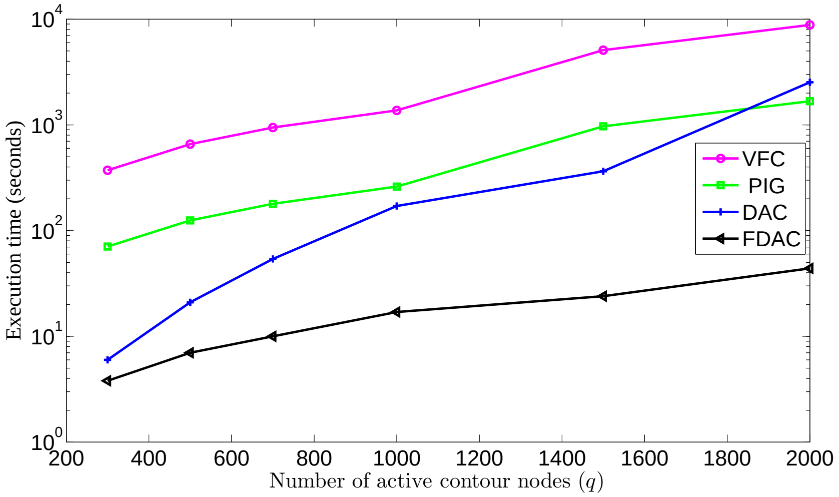


Fig. 5.16: The convergence rate of FDAC compared to DAC, VFC and PIG on a big synthetic star shape image of size 1970 × 1546 pixels. FDAC is found to be faster than other three methods with larger number of active contour vertices. A large number of active contour vertices are required to capture high curvature boundaries, and this phenomenon is presented in Fig. 5.3.

ages. On average, the convergence time of FDAC is found to be two orders of magnitude faster when compared to three state-of-the-art published methods.

In principle, the FDAC architecture is highly scalable to multiple dimensions. However, adding a dimension to FDAC completely changes the representation scheme (data structure) and complicates the computation of normals, curvature, and first and higher order derivatives. Further, performing resampling in 3D is a much more challenging task than in 2D. Therefore, the decoupled active surface (DAS), an extension of fast decoupled active contour from 2D to 3D, is developed and implemented in the next chapter.

Chapter 6

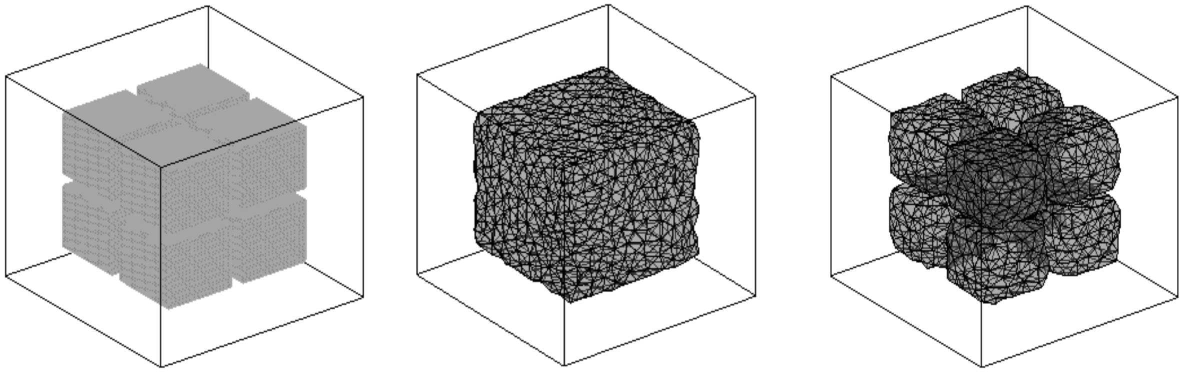
Decoupled Active Surface (3D)

The effectiveness of the DAC and FDAC to identify the boundary of a 2D object in the presence of noise and background clutter is demonstrated in Chapters 4 and 5. Besides the identification of 2D object boundaries, another important problem in computer vision is the identification of the surface of three-dimensional static object or similarly of two-dimensional objects over time.

As discussed in Chapter 2, active surface techniques [21–23] have been applied to such computer vision tasks. In active surface techniques, a deformable spline surface model is evolved based on the influence of internal and external (typically opposing) energies until the model converges to the desired surface. Active surface models [21–23, 39, 40, 54] are usually direct extensions of existing 2D active contour models to 3D active surface model.

As a result, similar to existing parametric active contour models (Section 4.4), current parametric active surface models tend to be computationally expensive (Section 5.5), difficult to implement [39] and not able to effectively identify the surface of a 3D object in the presence of noise, high curvature, and background clutter (Section 4.4). In contrast, non-parametric [35, 37, 39, 53, 114] methods are inherently scalable to multiple dimensions and are inherently able to identify multiple boundaries. However, the high computational burden and the inability to identify the continuous boundary [96] of a single broken object hinder the application of non-parametric active surfaces.

An example illustrating the tradeoff between a parametric and non-parametric active surface models is shown in Fig. 6.1. The non-parametric active surface model identifies



(a) Volumetric image (b) Parametric surface (c) Non-parametric surface

Fig. 6.1: Example of the suitability of parametric active surfaces over non-parametric active surfaces. (a) An image of a broken cube. (b) Parametric active surface finds a single connected surface of the broken cube. (c) Non-parametric active surface identifies eight separate surfaces.

eight separate surfaces (Fig. 6.1(b)) of a single broken cube, however, a parametric active surface model finds a single continuous boundary around the broken cube.

In tackling these issues, this chapter primarily develops a decoupled active surface (DAS) model, which extends the FDAC from 2D to 3D by modifying the FDAC representation scheme, to identify the surface of a static 3D object or volumetric tunnel of a sequence of moving 2D objects over time. The computational efficiency and outlier robustness of DAS are demonstrated in Section 6.6.

6.1 Challenges in Moving From 2D to 3D

The limitations of active surface models are discussed in Section 2.4. To demonstrate the primary issue of the current parametric active surface based methods, an example showing the convergence of (6.1) (same as (2.22))

$$\mathcal{F}_{int}(\mathbf{v}) + \mathcal{F}_{ext}(\mathbf{v}) = \varepsilon \frac{\partial \mathbf{v}}{\partial t}, \tag{6.1}$$

is shown for a synthetic volumetric cube with a spherical initial solution (black) in Fig. 6.2(a). For traditional representation schemes which map a 2D grid to the 3D surface, enforcing boundary conditions is a complicated task, as shown in Fig. 6.2. Using free boundary and free pole conditions in (6.1), a broken surface is generated (Fig. 6.2(b)). Similarly, a free pole condition generates holes at both poles of the converged surface (Fig. 6.2(c)). The use of constrained boundary and poles lead to a biased internal force which creates a non-uniform vertex motion, such that the vertices near the poles have a lower velocity compared to other vertices (Fig. 6.2(d)).

In summary, apart from the inherent 2D issues discussed in Section 2.4, the traditional active surface techniques have the following four additional problems.

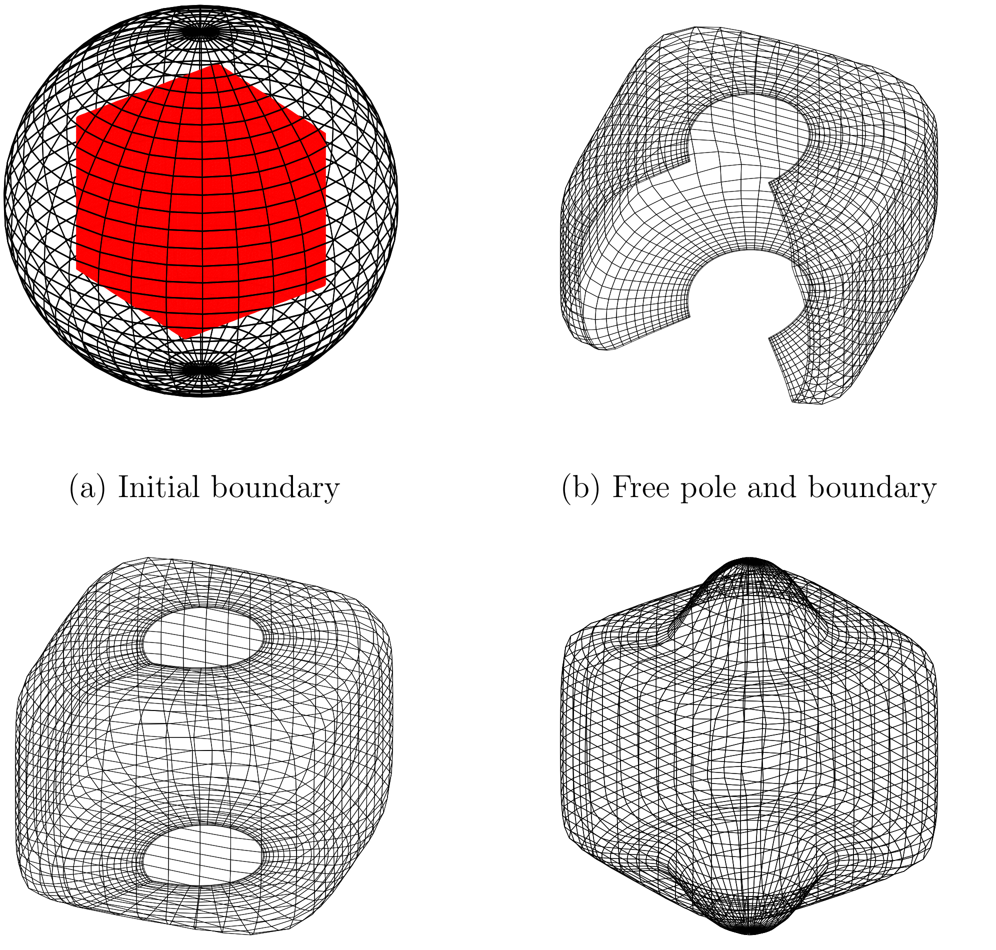
1. The conventional surface representation scheme described in (2.4) is not trivial to model complex geometries (Subsection 2.2.1) due to the many-to-one mapping problem.

2. Even in cases where the geometry can in principle be represented, the 3D-2D mapping creates a biased internal force, as shown in Fig. 6.2(d). This biased internal force makes the active surface model ineffective in the presence of outliers.

3. The traditional active surface is a direct extension of an active contour from 2D to 3D and uses an iterative gradient descent approach to solve (6.1). Such an iterative solver is slow and sensitive to local minima.

4. There exists a delicate relationship between the weights (parameters) contained inside the terms of (6.1).

Similar to parametric active contours (Section 2.4), recent parametric active surface methods, such as gradient vector flow (GVFS) [21], vector field convolution (VFC) [22] and Poisson inverse gradient (PIG) [54] have attempted to solve the initialization sensitivity and inability to converge towards concave regions problems by increasing the capture range of the traditional active surface. Typically these methods increase the capture range of the deformable model by diffusing the conventional gradient-based external force. However, these modifications are not able to overcome the problems of local minima and convergence speed. Detailed descriptions of several parametric and non-parametric active surface based boundary extraction techniques can be found in [96]. The extension of the FDAC from 2D to 3D is described next.



(a) Initial boundary

(b) Free pole and boundary

(c) Constrained boundary and free pole

(d) Constrained boundary and pole

Fig. 6.2: Three examples (b, c and d) illustrating the limitations of traditional, parameterized active-surface schemes. The algorithm start with an initial, parameterized spherical surface (a), seeking to converge to the synthetic cube. The partially-converged active surfaces after 500 iterations are shown in panels (b), (c) and (d). Parametric models have difficulties with boundaries (b) and singularities at poles (c,d).

6.2 From FDAC to DAS

Moving from FDAC in 2D to DAS in 3D is a non-trivial problem due to the necessity of a dedicated data structure to represent DAS. This is because in 2D space the neighboring vertices of a vertex in FDAC or DAC can be defined as its “previous” and “next” vertices. However, for DAS, such an assumption is not valid. Therefore, this section first describes a representation scheme suitable for DAS.

As shown in Fig. 6.3, a decoupled active surface $\mathbf{v}$ (2.5) is defined using conforming triangular meshes. These meshes are represented using triangular faces $T = \{t_k\}, k \in [1, t]$ and vertices $\mathbf{v} = \{v_j\}, j \in [1, q]$. The relationship between the faces and vertices are defined using two containers $\mathcal{TV}$ and $\mathcal{VT}$ such that:

$$\mathcal{TV}(t_k) = \{v_{ak}, v_{bk}, v_{ck}\} \tag{6.2}$$

and

$$\mathcal{VT}(v_j) = \{t_{dj}, t_{ej}, t_{fj} \cdots\}, \tag{6.3}$$

where $\mathcal{TV}$ and $\mathcal{VT}$ define links for moving from a face to vertices and from a vertex to faces, respectively. Given these two containers $\mathcal{TV}$ and $\mathcal{VT}$, the neighboring vertices of a vertex v_j can be defined as

$$\Omega(v_j) = \text{setdiff}(\text{unique}(\mathcal{TV}(\mathcal{VT}(v_j))), v_j), \tag{6.4}$$

where “setdiff” and ‘are two boolean operators. The “setdiff” operator is used to find the difference between two sets, and the “unique” operator is used to delete the repeated elements of a set. To demonstrate this representation scheme, an exemplar conforming triangular mesh based deformable model is illustrated in Fig. 6.3 using $q = 12$ number of vertices and $t = 20$ number of triangular faces. Given this triangular mesh based representation scheme, as illustrated in Fig. 6.4, the measurement, resampling and estimation steps of DAS are carried out in a similar manner as the FDAC. However, performing these three steps in 3D is difficult and need special consideration, as FDAC and DAS use different representation schemes. The description of these three steps, measurement (Section 6.3), resampling (Section 6.4) and estimation (Section 6.5) of DAS follows.

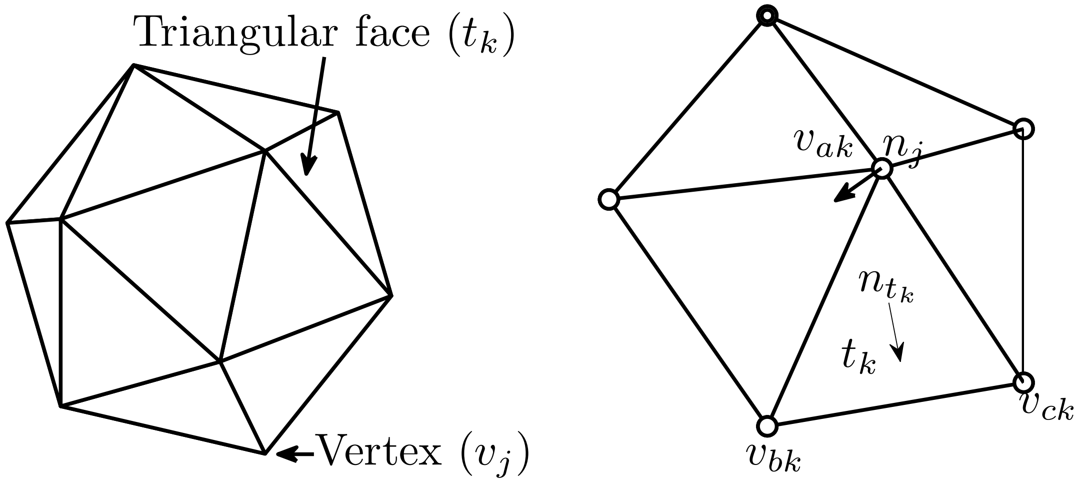


Fig. 6.3: Left, a spherical deformable model is represented using conforming triangles. Right, a portion of the left sphere is shown at a magnified scale for better visualization. The face and vertex normals are illustrated using arrows.

6.3 Step 1: Measured Surface Determination

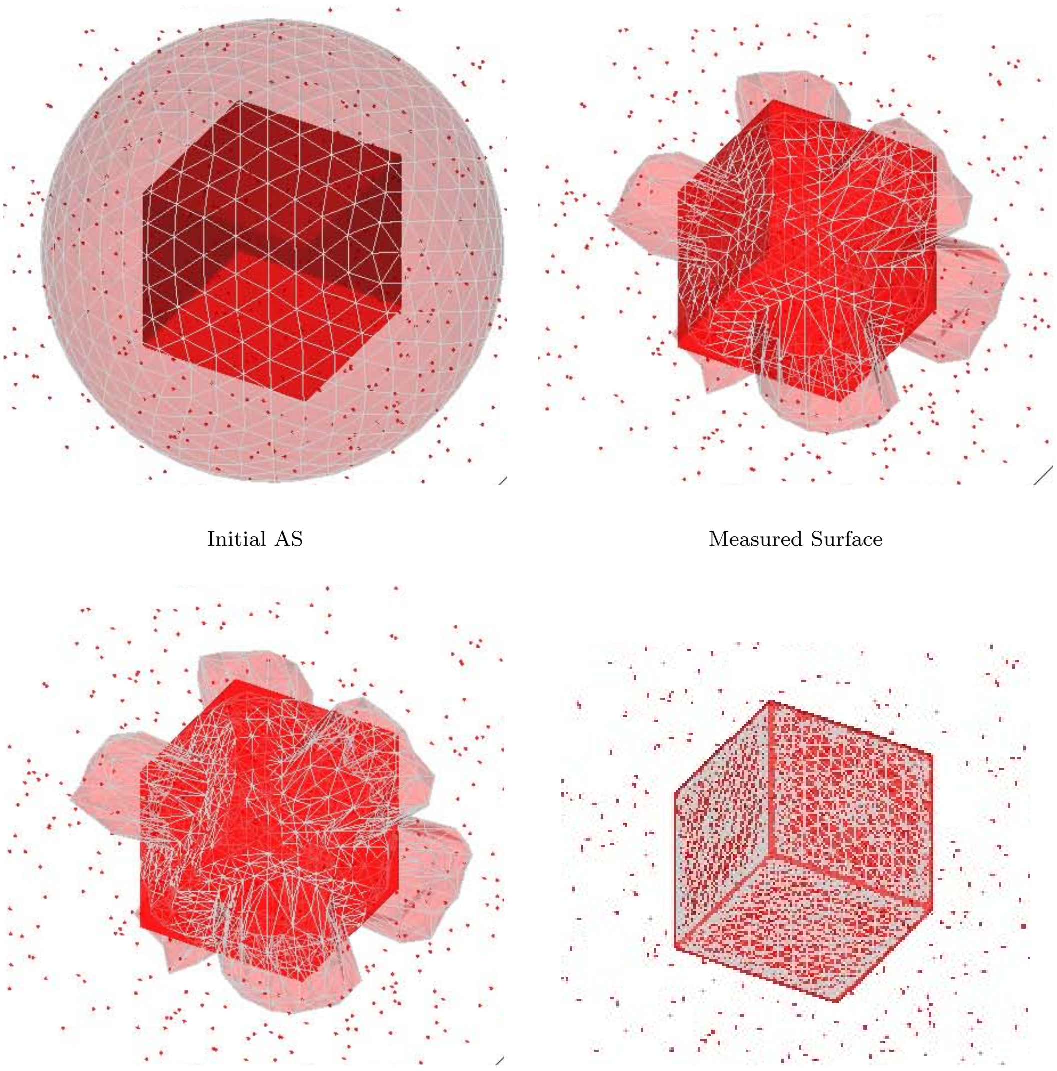
The iterative quasi-random search method developed in Section 5.2 can be applied to identify the measured surface of a 3D object by modifying the 2D trellis to accommodate for the 3D surface representation scheme. As described in Sections 4.1 and 5.2 the formation of the 2D trellis is initiated using a set of normals. However, computing the surface normals at the vertices of the 3D surface using (4.5) is not obvious. Therefore, as shown in the right panel of Fig. 6.3, the normals pointing to a face $n_{t_k}, k \in [1, \cdots, t]$ is first computed as

$$n_{t_k} = \frac{(v_{bk} - v_{ak}) \otimes (v_{ck} - v_{ba})}{|(v_{bk} - v_{ak}) \otimes (v_{ck} - v_{ba})|}, \tag{6.5}$$

and then a normal n_j pointing towards a vertex v_j

$$n_j = \frac{\sum\limits_{k \in \mathcal{VT}(v_j)} n_{t_k}}{c_j} \tag{6.6}$$

is computed by averaging the face normals n_{t_k} incident to the vertex v_j, where $|\cdot|$ and $\otimes$ are the norm and cross product operators, respectively and the term c_j represents the number of neighboring vertices of vertex v_j.



Initial AS

Measured Surface

Sampled surface

Converged surface

Fig. 6.4: The three important steps of DAS: for a given initial surface (left), gradients are measured, the resulting surface is resampled on the basis of curvature, and then smoothness prior is enforced into the resampled surface. Despite the presence of noise in this example, the proposed iterative approach converged well to the desired cube, right.

Given these vertex normals $n_j, j \in [1, \cdots, q]$, the 3D trellis is created using the formulation developed in Section 4.1. Then, the iterative quasi-random search (IQRS) method developed in Section 5.2 is employed to find the measured surface as discussed below.

This development follows the IQRS formulation proposed in Section 5.2, the states $w_j^m, j \in [1, q]$ having maximum probability along the normals $n_j, j \in [1, \cdots, q]$ are first computed as:

$$w_j^m = \underset{w_j \in [-\frac{u}{2}, \cdots, \frac{u}{2}]}{\arg\max} \left(P\left(\psi(w_j)\right)\right), j \in [1, \cdots, q]. \tag{6.7}$$

Second, following the description of the IQRS approach of the FDAC (Section 5.2), a spatial inter-neighboring constraint is enforced iteratively to overcome unnecessary peaky surface vertices. Given this likeliness constraint, the state of the normals $w_j, j \in [1, \cdots, q]$ are updated as:

$$w_j^m = \underset{w_j}{\arg\max} \left(P\left(\psi\left(w_j\right) | w_j, \Omega\left(w_j^m\right)\right)\right), \tag{6.8}$$

which is equivalent to

$$w_j^m = \underset{w_j}{\arg\max} \left(P\left(\psi\left(w_j\right) | w_j\right) P\left(w_j, \Omega\left(w_j^m\right)\right)\right), \tag{6.9}$$

where

$$P\left(w_j, \Omega\left(w_j^m\right)\right) = \frac{1}{Z_b} \exp\left(- \sum_{v \in \Omega\left(v_{w_j^m}\right)} \left\| v_{w_j} - v \right\| \right). \tag{6.10}$$

The term Z_b is a normalizing constant to make $P\left(w_j, \Omega\left(w_j^m\right)\right)$ a probability distribution functions for normal j. The pseudo-code for IQRS is provided in Algorithm 5.

The converged boundary $\mathbf{v}^m$ using IQRS is assigned as the measured boundary. As described in Section 4.3, a non-stationary prior is essential in the update step to capture high curvature surfaces. The generation of the non-stationary prior using curvature based mesh resampling (Section 6.4) and the estimation technique (Section 6.5) are provided next.

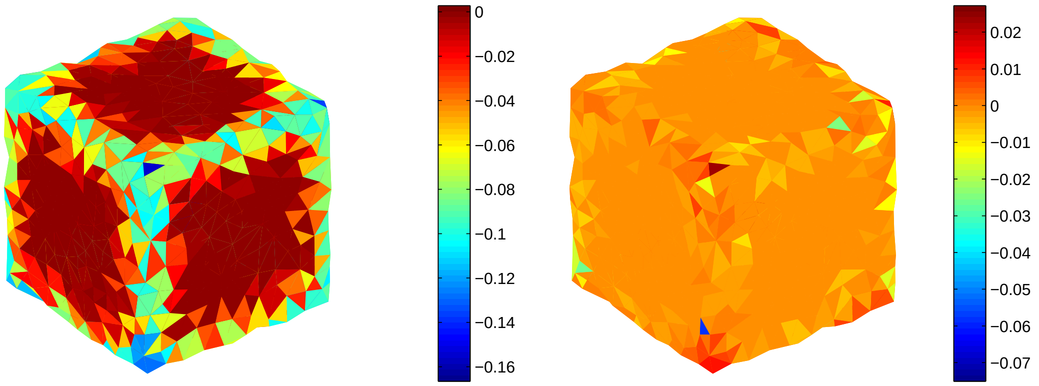


Fig. 6.5: The principal and Gaussian curvatures of a cubic surface.

6.4 Step 2: Generation of the Non-Stationary Prior in 3D

High curvature regions (corners, edges) of a surface can be captured by relaxing the prior. This has been accomplished in Sections 4.2 and 5.3 via an importance sampling step in 2D, placing a greater density of samples near regions of high curvature. However, computing the surface curvature and performing the importance sampling in 3D are two challenging tasks. The description of these two tasks follows.

6.4.1 Curvatures in 3D

The curvature $\boldsymbol{\kappa}$ of a parametric 2D curve $v(s) = [x(s), y(s)], s \in [0, 1]$ is defined as the rate of change of tangent angle with respect to the arclength s. However for a 3D surface the curvature at a particular vertex varies as the plane through the normal at that point changes, therefore there is no unique definition of surface curvature. As shown in Fig. 6.5, the curvature of a 3D surface is usually defined using principal and Gaussian curvatures. However, for DAS's purposes an exact definition of curvature is not needed; rather what is needed is some measure of triangulation error, such that additional triangles are placed near edges and corners. For this purpose, the pseudo curvature κ_j can be computed first by using a least squares algorithm to fit a plane to the first-order neighboring vertices $\Omega(v_j)$, followed by the computation of the perpendicular distance from surface vertex v_j to the least squares fitted plane. The mathematical formulation to compute the pseudo curvature $\kappa_j, j \in [1, \cdots, q]$ follows.

Let a plane $a_1x + a_2y + a_3z - 1 = 0$ be approximated to fit $\Omega(v_j)$, the neighboring vertices of the vertex $v_j = [x_j, y_j, z_j]$. In this problem, the number of vertices (linear equations) are more than the number of variables (a_1, a_2, a_3). Hence a linear least squares algorithm is employed to find the necessary plane. The algorithm assumes the set of neighboring vertices $\Omega(v_j) = [v_{bk}, v_{ck}, \cdots]^T$ of the vertex v_j lies on the hypothetical plane $a_1x + a_2y + a_3z - 1 = 0$. The plane that passes through $\Omega(v_j)$ can be defined using a set of linear equations as:

$$\begin{bmatrix} x_{bk} & y_{bk} & z_{bk} \\ x_{ck} & y_{ck} & z_{ck} \\ \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \vdots \end{bmatrix}. \tag{6.11}$$

Using vector and matrix notation these set of linear equations can be rewritten as

$$\Pi\, \mathcal{B} = \mathbf{1}, \tag{6.12}$$

where $\mathcal{B} = [a_1, a_2, a_3]^T$. Using linear least square fitting algorithm $\mathcal{B}$ is given by:

$$\mathcal{B} = \left(\Pi\, \Pi^T\right)^{-1}\ \Pi^T \mathbf{1} \tag{6.13}$$

The perpendicular distance of the vertex v_j from this plane is given by:

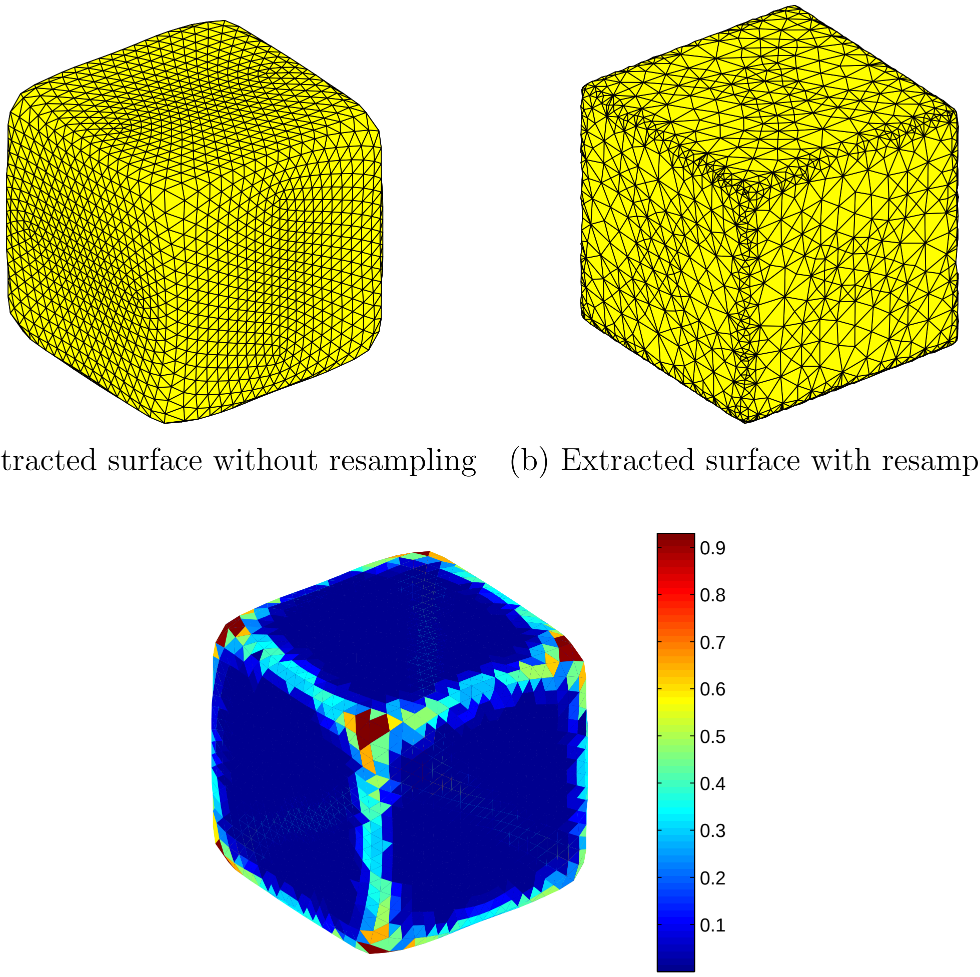
$$\kappa_j = \frac{a_1x_j + a_2y_j + a_3z_j - 1}{\sqrt{a_1^2 + a_2^2 + a_3^2 + 1}}, j \in [1, \cdots, q] \tag{6.14}$$

The pseudo curvature of a synthetic cube computed using (6.14) is demonstrated in Fig. 6.6(b). Having computed $\boldsymbol{\kappa}$ at all vertices, the curvature based mesh resampling scheme is provided next.

6.4.2 Surface Resampling

The surface resampling step of DAS is necessary for three primary reasons.

1. To generate a curvature based non-stationary prior to capture high curvature surfaces.



(a) Extracted surface without resampling (b) Extracted surface with resampling

(c) Curvature of the cube

Fig. 6.6: The resampling scheme. The top left cube (a) is iterated without resampling, but ends up with an excess of grid points in flat regions, and possibly a deficiency in regions of high curvature. The top right (b) cube shows the mesh after resampling, based on the curvature as shown in the bottom panel (c).

Algorithm 9 Function MeshResample1

```
while $j \leq q$ do
    Compute the pseudo curvature $\kappa_j$ using (6.14);
    if $\kappa_j > \kappa_{th}$ then
        Get the edges $e(j)$ that are connected to the vertex $v_j$;
        Get the longest edge length $l_l$ and the index $i$ corresponding to $l_l$ from the length $l(e_i(j))$ of all edges;
        if $l_l > l_{th}$ then
            As shown in Fig. 6.8(c), bisect the edge $e_i(j)$ by inserting a new vertex, and then update $\mathcal{VT}$ and $\mathcal{TV}$;
        end if
    end if
    $j = j + 1$;
end while
```

2. To lower the computational complexity by reducing the total number of surface vertices by fewer samples in smooth regions.

3. To maintain numerical stability of the surface evolution process by avoiding the formation of unwanted triangles.

To achieve the first and second goals, Algorithm 9 generates the resampled set of vertices using the curvature $\boldsymbol{\kappa}$ such that faces with curvature exceeding a threshold κ_{th} are considered for subsampling, and then resampling is performed along the longest edge of that face. To achieve the third goal, DAS refines two types of problematic triangles (thin and fat) that are generated during the surface evolution process.

As shown in Fig. 6.7(a), the triangles whose longest to the shortest edge length ratio are Th_1 and the largest to second largest edge length ratio are between Th_2 to Th_3 are termed as tall triangles. Other triangles that are not tall triangles but have the length of the longest edge length greater than Th_4 are considered as fat triangles. A fat triangle is illustrated in Fig. 6.8(a). Due to excessively large segment length, both tall and fat triangles can lead the deformable model to an unstable state, if not refined at an earlier iteration. Hence, the thin and fat triangles are refined. The refining process is carried out using two operations: triangle splitting and mesh closing as described in Algorithms 10

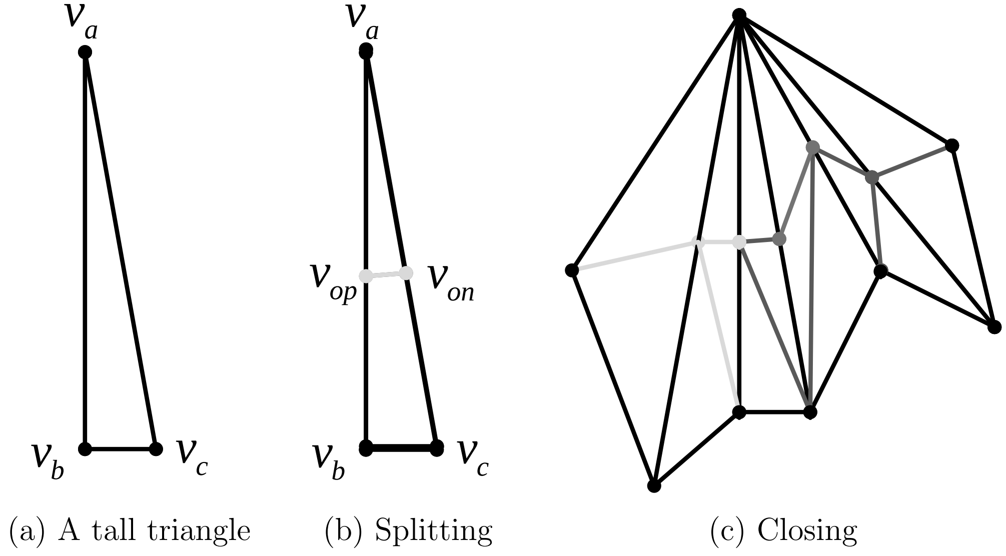


Fig. 6.7: Demonstration of the splitting and closing operations of a tall triangle. Left, a tall triangle. Middle, splitting of a tall triangle creates two open vertices (v_{op} and v_{on}). Right, backward (light grey) and forward (dark grey) closing operations are performed to close the open vertices. The original surface vertices and segments are marked in black, while the newly inserted vertices and segments are marked in grey.

and 11, and shown in Figs. 6.7 and 6.8. As shown in Figs. 6.7(b) and 6.8(b), the splitting of a tall triangle generates two open vertices and the splitting of a fat triangle generate an open vertex. Where, an open vertex is a vertex which creates a hole in a boundary if not tied to its neighboring vertices appropriately.

The splitting and closing operations of a tall triangle are demonstrated in Fig. 6.7. Where, the longest and the second longest edges of the tall triangle are bisected by inserting (middle panel) two new vertices (v_{on} and v_{op}) and then these two open vertices are closed using a backward and a forward closing operations (right panel). Similarly, the splitting and closing operations of a fat triangle are demonstrated in Fig. 6.8. Where the largest edge of the fat triangle is bisected (middle) by inserting a new vertex (v_o) and then the open vertex is closed using a closing operation (right). Finally, The algorithm to refine all fat and thin triangles is described in Algorithm 12.

Algorithm 10 Function Split(t_i)

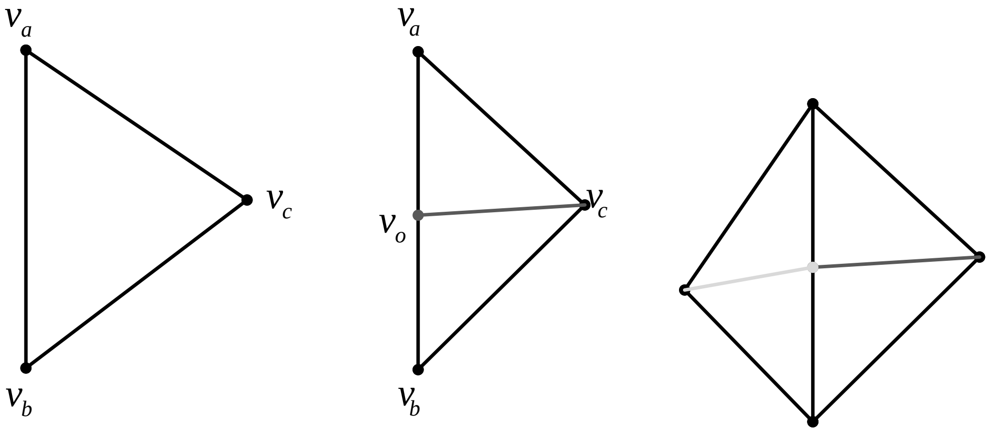
```
if $t_i$ is a thin triangle then
    As shown in Fig. 6.7(b), insert two new vertices one at the middle of the longest
    ($oep$) and another at second longest $oen$ edges of the triangle $t_i$, respectively;
    Update the containers $\mathcal{TV}$ and $\mathcal{VT}$;
    CloseMesh(oep);
    CloseMesh(oen);
else
    if $t_i$ is a thick triangle then
        As shown in Fig. 6.8(b), insert a new vertex $ov$ at the middle of the longest edge
        $oe$ of the triangle $t_i$;
        Update the containers $\mathcal{TV}$ and $\mathcal{VT}$, TRAVERSED($t_k$)=1;
        CloseMesh(oe);
    end if
end if
```

On the other hand, reducing the number of vertices is a far more difficult operation as such a process needs re-triangulation [34, 115]. Therefore, DAS is initialized with a deliberately small number of vertices, with further vertices added as needed, but never removed.

Finally, the DAS's mesh resampling scheme is illustrated using a synthetic cube in Fig. 6.6. More samples near the corners and edges of the cube can be observed in 6.6(b).

6.5 Step 3: Bayesian Estimation in 3D.

Two different techniques to fuse the non-stationary prior with the measured boundary using a Bayesian estimation approaches are demonstrated in Sections 4.3 and 5.4. The primary focus of this section is to propose a computationally and storage efficient conjugate gradient technique, and an approach to compute the prior constraints Q in 3D to solve the Bayesian estimation problem. Based on the formulation developed in Section 5.4 (pp. 84), an estimate of each component $[\mathbf{x}^m, \mathbf{y}^m, \mathbf{z}^m]$ of the measured surface $\mathbf{v}^m$ can be expressed



(a) A fat Triangle (b) Splitting (c) Closing

Fig. 6.8: Demonstration of the splitting and closing operation of a fat triangle. Left, a fat triangle. Middle, splitting of a fat triangle creates an open vertex (v_o). Right, a closing operation is performed to close the open vertex. The newly inserted vertex and edges are marked in grey.

as:

$$\mathcal{H}\mathbf{x}_\theta = \mathbf{b}_x, \quad \mathcal{H}\mathbf{y}_\theta = \mathbf{b}_y, \quad \mathcal{H}\mathbf{z}_\theta = \mathbf{b}_z, \tag{6.15}$$

where

$$\mathbf{x}_\theta = \left(\hat{\mathbf{x}} - \boldsymbol{\mu}_x\right), \quad \mathbf{y}_\theta = \left(\hat{\mathbf{y}} - \boldsymbol{\mu}_y\right), \quad \mathbf{z}_\theta = \left(\hat{\mathbf{z}} - \boldsymbol{\mu}_z\right), \tag{6.16}$$

$$\mathbf{b}_x = R^{-1}\left(\mathbf{x}^m - \boldsymbol{\mu}_x\right), \quad \mathbf{b}_y = R^{-1}\left(\mathbf{y}^m - \boldsymbol{\mu}_y\right), \quad \mathbf{b}_z = R^{-1}\left(\mathbf{z}^m - \boldsymbol{\mu}_z\right) \tag{6.17}$$

and

$$\mathcal{H} = (R^{-1} + Q), \tag{6.18}$$

where R and Q are the measurements uncertainties and prior constraints matrices. The values of $\mathbf{x}_\theta$, $\mathbf{y}_\theta$ and $\mathbf{z}_\theta$ can be computed using the conjugate gradient technique proposed in Algorithm 7.

To solve (6.15) using conventional conjugate gradient method proposed in Algorithm 7 involves a matrix ($\mathcal{H}$) and a vector $\mathbf{b}$ multiplication and such a multiplication needs large computational as well as storage resources for solving a 3D problem. For example, a moderate size 3D object requires $q = 5000$ vertices to represent its surface, and as a result

Algorithm 11 Function CloseMesh(e)

```
if $t_k$ is a thin triangle then
    Split($t_k$)
else
    if $t_i$ is a fat triangle then
        Close the mesh as shown in Fig. 6.8(d), TRAVERSED($t_k$)=1;
        Update the containers $\mathcal{TV}$ and $\mathcal{VT}$;
    end if
end if
```

Algorithm 12 Function MeshResample2

```
Assign TRAVERSED = {0};
for $k = 1 : t$ do
    if TRAVERSED($t_k$)== 0 then
        Split($t_k$), TRAVERSED($t_k$)=1;
    end if
end for
```

the size of $\mathcal{H}$ becomes 5000×5000. However the prior constraints (Q) is a sparse circulant matrix and the noise covariance (R) is a diagonal matrix. Hence, $\mathcal{H} = (R^{-1} + Q)$ is also a sparse circulant matrix. As a result, $\mathcal{H}$ can be completely and sparsely specified by only two vectors: **h** and **r** as shown below

$$\mathcal{H} = \text{circulant}(\mathbf{h}) + \text{diag}\left(\frac{1}{\mathbf{r}}\right), \tag{6.19}$$

where **r** is the diagonal of the matrix R and **h** is a kernel of the prior constraints Q. The measurements uncertainty matrix R ($r_j, j \in [1, \cdots, q]$) is computed using (4.33).

Next, the goal is to compute Q (**h**) on an irregular triangular mesh. As described in Subsection 2.2.2 (pp. 12), in the context of traditional active contour and surface, the internal energy (prior) is defined using an elastic (first order derivative) and thin-plate (second order derivative) energies. But, the first and higher order derivatives are not well defined for an irregular triangular mesh. However, a true definition of derivatives is not necessary for practical discrete implementation; instead a technique to compute the penalty on the slope and the curvature of a curve/surface is required.

Algorithm 13 $[\mathbf{x}_\theta]$ = Function MCG($\mathbf{b}$)

```
$\epsilon = 10^{-10}$, $\mathbf{x}_0 = \mathbf{b}$, $\boldsymbol{\varpi}_0 = \mathbf{x}_0 - \boldsymbol{\varphi}(\mathbf{x}_0)$, $\mathbf{o}_0 = \boldsymbol{\varpi}_0$, $k = 0$;
while $|\boldsymbol{\varpi}_k| > \epsilon$ do
  $\boldsymbol{\iota} = \boldsymbol{\varphi}(\mathbf{o}_k)$
  $\boldsymbol{\lambda}_k = \frac{\boldsymbol{\varpi}_k^T \boldsymbol{\varpi}_k}{\mathbf{o}_k^T \boldsymbol{\iota}}$, $\mathbf{x}_{k+1} = \mathbf{x}_k + \boldsymbol{\lambda}_k \mathbf{o}_k$, $\boldsymbol{\varpi}_{k+1} = \boldsymbol{\varpi}_k - \boldsymbol{\lambda}_k \boldsymbol{\iota}$;
  $\boldsymbol{\rho}_k = \frac{\boldsymbol{\varpi}_{k+1}^T \boldsymbol{\varpi}_{k+1}}{\boldsymbol{\varpi}_k^T \boldsymbol{\varpi}_k}$, $\mathbf{o}_{k+1} = \boldsymbol{\rho}_k \mathbf{o}_k + \boldsymbol{\varpi}_{k+1}$, $k = k + 1$;
end while
$\mathbf{x}_\theta = \mathbf{x}_k$;
```

To compute these two penalties, DAS proposes a pseudo definition of the second Δ^2 and the fourth Δ^4 order synthetic derivatives of a discrete entity x as:

$$\Delta^2 x_j = \frac{1}{c_j} \sum_{x \in \Omega(x_j)} (x - x_j) \tag{6.20}$$

and

$$\Delta^4 x_j = \frac{1}{c_j} \sum_{\Delta^2 x \in \Omega(\Delta^2 x_j)} \left(\Delta^2 x - \Delta^2 x_j\right), \tag{6.21}$$

where $\Omega\left(x_j\right)$ and $\Omega\left(\Delta^2 x_j\right)$ represent the x and $\Delta^2 x$ values of the neighboring vertices of the vertex v_j, and the term c_j representees the number of neighboring vertices of the vertex v_j.

Essentially, the ultimate goal is to estimate $\mathbf{x}_\theta$, $\mathbf{y}_\theta$ and $\mathbf{z}_\theta$. A conjugate gradient (CG) approach is proposed in Section 5.4 (pp. 85) to estimate these quantities, where the CG method first computes the large matrix H using Algorithm 6 (pp. 85) and then uses the pre-computed $\mathcal{H}$ to compute $\mathcal{H}\mathbf{b}$. However, this product $\mathcal{H}\mathbf{b}$ can be computed efficiently without unnecessarily storing the large matrix $\mathcal{H}$ by using a modified CG technique as described in Algorithm 13.

The modified CG method uses a function $\boldsymbol{\varphi}(\mathbf{b})$ (Algorithm 14) to dynamically compute the product $\mathcal{H}\mathbf{b}$ without storing $\mathcal{H}$. Essentially, for each vertex $(v_j, j \in [1, \cdots, q])$ the Algorithm 14 generates a non-zero band of the matrix $\mathcal{H}$ as described in (6.19) and computes the product $\mathcal{H}\mathbf{b}$ for the corresponding vertex using (6.22).

The modified algorithm to estimate the components $[\mathbf{x}, \mathbf{y}, \mathbf{z}]$ of the measured boundary $\mathbf{v}^m$ is presented in Algorithm 13. Finally, the three steps of DAS are provided in Algorithm 15.

Algorithm 14 $[\boldsymbol{\iota}] = \boldsymbol{\varphi}(\mathbf{x})$

```
for $j = 1 : q$ do
    Compute $\Delta^2 x_j$ using (6.20) and $\Delta^2 x_j$ using (6.20);
    Compute
        $$\iota_j = \alpha\Delta^2 x_j + \beta\Delta^4 x_j + \frac{x_j}{r_j}; \qquad (6.22)$$
end for
```

Algorithm 15 $[\mathbf{v}^c]$ = Function DAS($\mathbf{v}_0$)

```
$k = 0$, $\mathbf{v}_k^e = \mathbf{v}_k$, $\epsilon = 10^{-4}$, $ASD = \infty$;
while ASD $> \epsilon$ do
    $k = k + 1$, $\mathbf{v}_k^m$ = IQRS($\mathbf{v}_{k-1}^e$, $\epsilon$);
    $\mathbf{x}_k^e$ = MCG($\mathbf{x}_k^m$), $\mathbf{y}_k^e$ = MCG($\mathbf{y}_k^m$), $\mathbf{z}_k^e$ = MCG($\mathbf{z}_k^m$), $\mathbf{v}_k^e = [\mathbf{x}_k^e, \mathbf{x}_k^e, \mathbf{z}_k^e]$, ASD = GetASD($\mathbf{v}_k^e, \mathbf{v}_{k-1}^e$);
end while
$\mathbf{v}^c = \mathbf{v}_k^e$;
```

6.6 Testing and Results

The advanced capabilities of the DAC and the FDAC to segment 2D objects in the presence of noise and background clutter have been demonstrated in Sections 4.4 and 5.5. The DAS extends FDAC concepts to 3D and validates its claims using one natural (ES, Fig 6.11), four synthetic 3D volumetric images (AS, BS, CS and DS, Figs 6.9 and 6.10) and two 2D moving image sequence (FS and GS, Figs 6.12 and 6.13). The seven test cases involve typical characteristics of high curvature and noise. The proposed DAS is compared to a level set based hybrid [114] (LSHYB) active surface method. The code for LSHYB is downloaded from [116].

LSHYB integrates the object boundary and region information and uses a level set frame work to find the boundary of an object. LSHYB is considered as an effective hybrid active surface technique [114]. Therefore, the performance of DAS is compared to LSHYB.

All experiments are performed on a P4 Intel 2.4Ghz processor, 1Gb RAM using Matlab. The parameters for LSHYB are adopted from [114]. For DAS the parameters: $\kappa_{th} = 1$, $l_{th} = 8$, $\alpha = 0.1$ and $\beta = 0.01$ are used for all test cases. DAS is initialized with a spherical initial surface and LSHYB is initialized with a zero level set.

6.6.1 Qualitative Segmentation Result

A standard established 3D database for quantitative error analysis is not available and creating manual ground truth for 3D surfaces is a non-trivial task. Therefore, a qualitative analysis of the surface identification performance of DAS and LSHYB is provided. The DAS's ability to identify the surface of 3D volumetric images is presented in Figs. 6.9, 6.10, 6.11, 6.12 and 6.13.

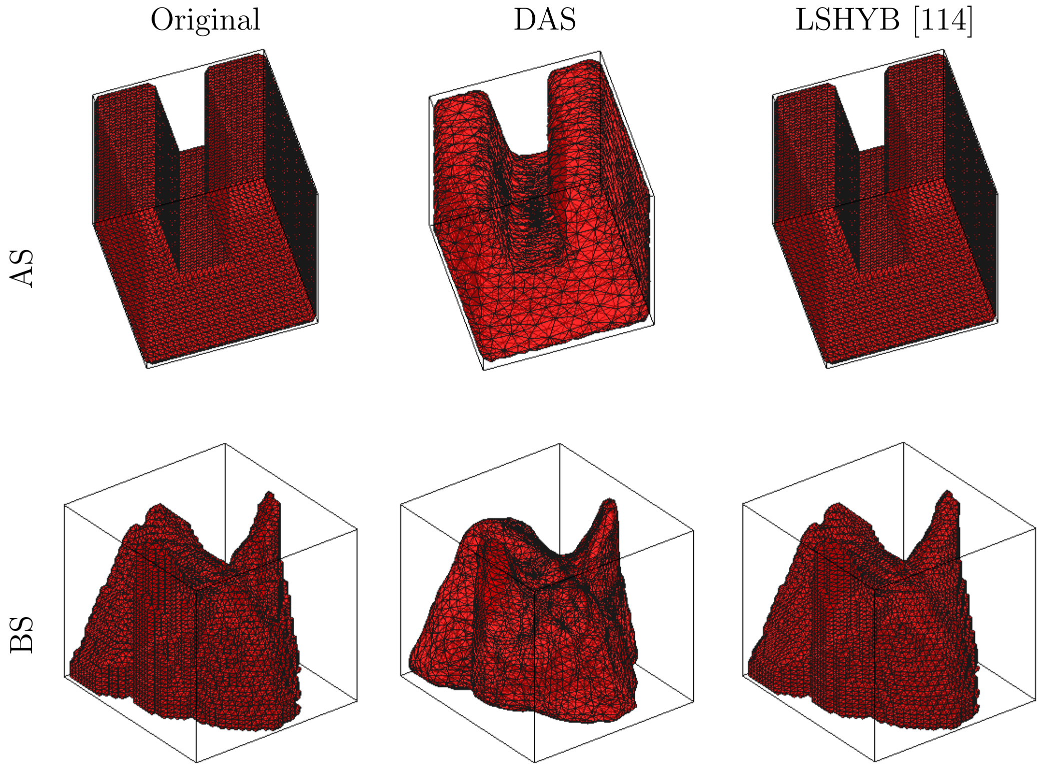


Fig. 6.9: Surface identification accuracy of DAS and LSHYB. The test is conducted on two surfaces: a U-shape (AS; top left), and a Bunny (BS, bottom left). Both DAS and LSHYB segment these two objects successfully.

Fig. 6.9 compares the performance of DAS (middle column) and LSHYB (right column). Non-parametric methods are better in capturing high curvature regions compared

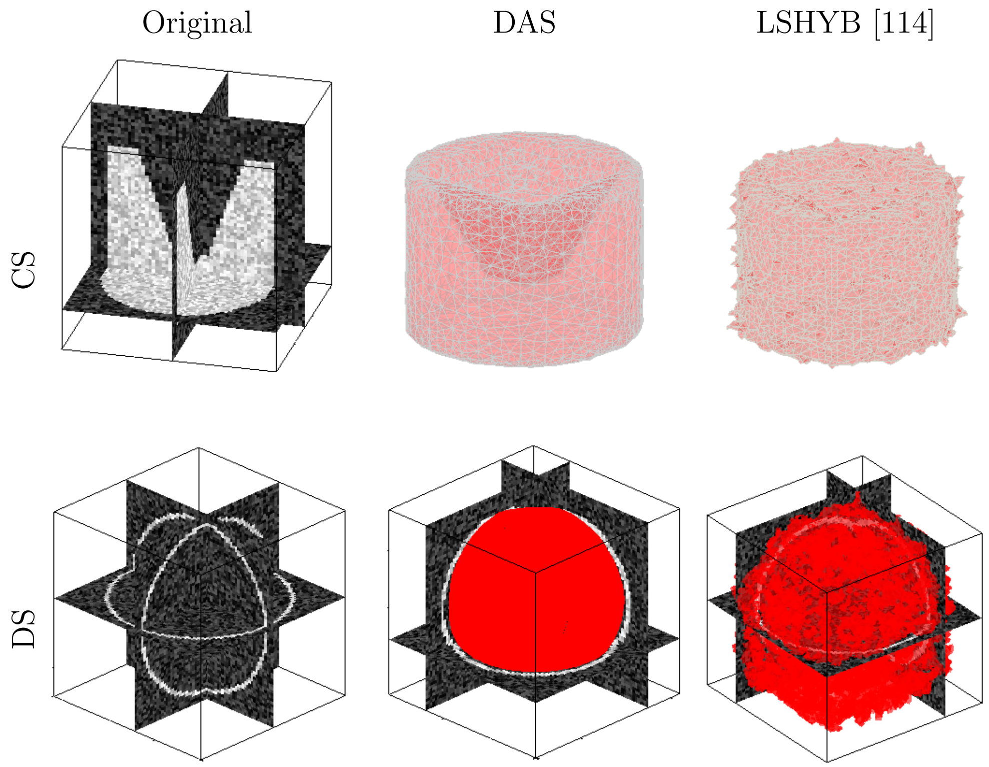
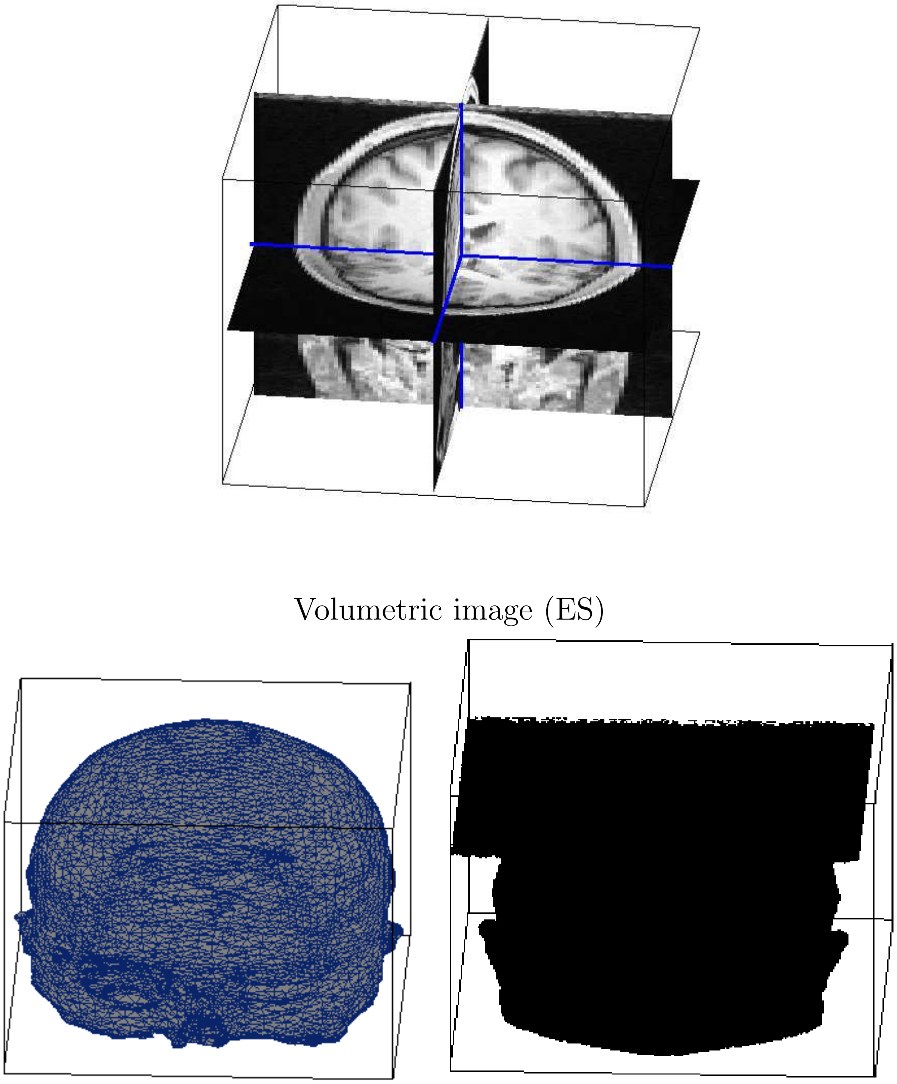


Fig. 6.10: The DAS successfully identified the outer surface of a hollow cylinder (top) and a sphere (bottom) in the presence additive Gaussian noise ($\sigma = 0.4$). As expected, LSHYB performed poorly in the presence of noise.

to parametric methods, therefore, LSHYB, a non-parametric method, identifies the high curvature boundaries of images AS and BS.

Though, DAS is a parametric model, it successfully identifies the high curvature boundaries of the images AS and BS using separated measurement and prior steps, and also by enforcing a lower force in high curvature regions. The high-curvature tolerance of DAS is illustrated most clearly in the sharpness of the top and bottom surface edges in image (AS).

The performance of DAS and LSHYB to identify the outer surface of a human head in a volumetric MR image is presented in Fig. 6.11. DAS successfully identifies high curvature regions (e.g., the ears, nose and eyes) in the MR image. As with non-parametric methods, LSHYB poorly identifies the outer surface of the human brain in the presence of the noise.



Volumetric image (ES)

DAS

LSHYB [114]

Fig. 6.11: Image (ES): a human MRI scan. The DAS successfully identified the outer surface of the MRI of the brain. The DAS's ability to capture high curvature regions (ears and nose) can be observed. LSHYB poorly identified the outer surface of the human MRI due to the presence of noise.

Finally, the ability of DAS to identify the tunnels of two walking men in two spatio-temporal sequences [117] are demonstrated in Figs. 6.12 and 6.13, respectively. In both cases, DAS successfully identified the human walking tunnels.

6.6.2 Robustness to Noise

The robustness of the decoupled deformable model in 2D has been demonstrated in Chapters 4 and 5 using illustrative examples. This chapter shows the ability of DAS (a decoupled deformable model in 3D) to identify the surface of noisy synthetic 3D images. The effectiveness of DAS to perform in the presence of noise (Gaussian, $\sigma = 0.4$) is demonstrated using two synthetic images (CS and DS) in Fig. 6.10 and a real measured human tomogram (ES) in the top panel of Fig. 6.11.

DAS successfully identifies the outer surfaces of the volumetric images in the presence of noise. In contrast, LSHYB (a non-parametric active surface method) fails to identify the true surfaces for all noisy volumetric images. As non-parametric methods do not use a parametric model, therefore, non-parametric methods are typically more sensitive to noise compared to parametric methods. However, non-parametric methods are insensitive to initial solution and inherently handle multiple topology.

6.6.3 Convergence Speed

The convergence time of DAS compared to LSHYB [114] is presented in Table 6.1. The computational complexity of LSHYB (a non-parametric method) is a function of number of voxels. In contrast, the computational complexity of DAS (a parametric method) is a function of number of active surface vertices. Further, the mesh resampling step of DAS reduces the number of vertices significantly by putting fewer vertices near smooth regions. Therefore, as shown in Table 6.1, the convergence time of DAS is significantly lower than that of LSHYB [114].

6.6.4 Summary of DAS

A decoupled active surface model based on the extension of the FDAC from 2D to 3D is proposed and implemented for identifying the surface of a volumetric 3D object and the

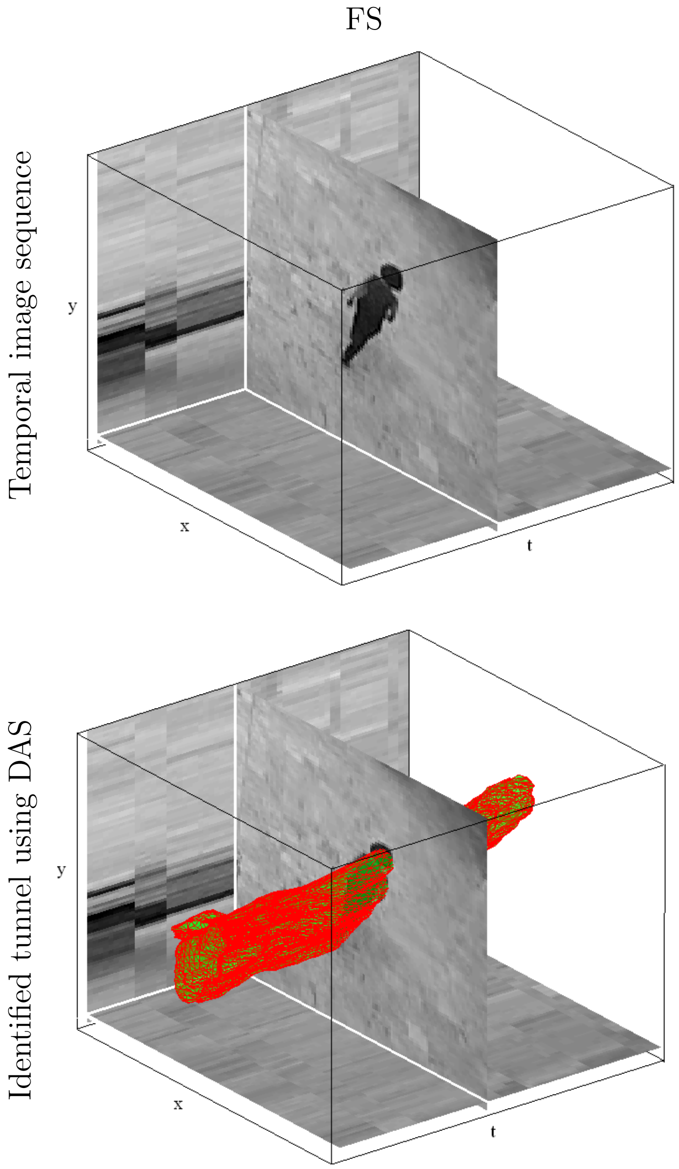


Fig. 6.12: Image GS: Image sequence of a moving 2D object (PETS 2007 dataset [117]), namely a man walking (top panel). The DAS successfully identified the desired tunnel traced by the walking man (bottom panel). For clarity, the tunnel and three slices in $x-y$, $y-t$, and $x-t$ planes are presented in the bottom panel. The black object is the walking man.

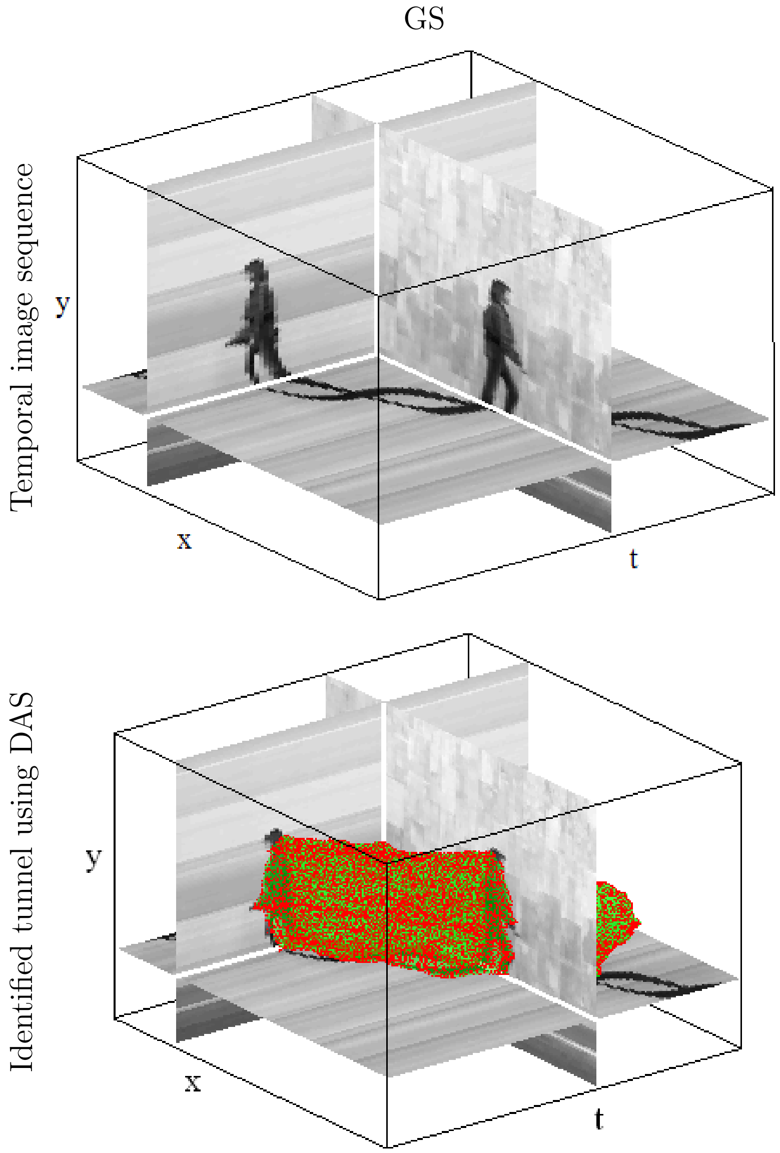


Fig. 6.13: Image HS [18]: Image sequence of a moving 2D object, namely a man walking (top panel). At first sight, it seems two people are walking. However, only one man is walking along the time axis (t-slice). The other look alike man along the x-axis (x-slice) is the cross section of the human as it appears along time. The tunnel is shown using green (face color) and red (edge color) colors.

Table 6.1: Comparison table showing execution time in seconds for the surfaces in Figs. 6.9, 6.10 6.11, 6.12 and 6.13 comparing the DAS and the LSHYB [114]. The DAS is more than two order of magnitude faster than LSHYB.

	LSHYB [114]		DAS		
	No. of vertices q	ET	Initial no. vertices	Final no. vertices	ET
AS	14900	1980	642	1961	323
BS	21284	2383	642	3684	187
CS	18910	2542	642	2432	297
DS	78888	3323	1281	1842	123
ES	499044	15623	1281	4987	315
FS	-	-	2562	7096	512
GS	-	-	2562	9748	539

tunnel of a moving 2D object. The dramatic computational gain and better segmentation performance of DAS compared to one published parametric active surface based method have been demonstrated in this chapter. The better segmentation performance and success of DAS in the presence of noise and high curvature is due to the use of the novel concept of separating the prior from measurements and the incorporation of a non-stationary prior using a mesh resampling scheme.

Chapter 7

Summary and Future DDM Research

This chapter summarizes the work of this thesis and provides further directions to extend the work described in this thesis.

7.1 Summary of DDM

Decoupled deformable model (DDM), a novel approach to minimizing the total energy of the active contours/surfaces by separating the prior from the measurement has been designed and implemented. First, the concepts of the DDM is applied in 2D to develop a decoupled active contour (DAC) to precisely identify the boundary of a single object in the presence of noise and background clutter. To carry out the boundary extraction task in 2D, each iteration of the DAC is performed in the following three steps. i) A Viterbi search is performed to find measured boundary from the image gradients. ii) An importance resampling step is performed to generate a non-stationary prior to capture high curvature regions. iii) A Bayesian estimator is applied to update the measured boundary by incorporating prior shape constraints.

The objectives of DAC are validated experimentally on noisy, cluttered natural and synthetic images, and on an established database compared to three published methods. DAC is demonstrated to be robust to noise, requires no parameter tuning, is able to capture high curvature regions, and is insensitive to initialization. The convergence rate of DAC is two orders of magnitude faster than the compared methods.

Second, a fast decoupled active contour (FDAC) is proposed to reduce computational burden and to enhance the scalability of DAC to 3D by using more advanced measurement, resampling and estimation schemes than DAC. Tests comparing FDAC to three published methods have been performed using both natural and synthetic images. FDAC's computational speed is approximately two to three times faster than DAC's and two orders of magnitude faster with comparable or reduced error when compared to two other methods. FDAC's dramatic reduction in computational load allows the use of additional vertices to improve identification in high curvature boundary regions. The computational gain of FDAC over DAC stems from the use of an iterative quasi-random search approach to identify boundary-like features, a uniform sampling on the integral of the absolute value of the curvature of the measured boundary to generate a non-stationary prior, and an efficient conjugate gradient technique to assert the prior model.

Third, a decoupled active surface (DAS) is designed by implementing the three steps of FDAC in 3D. First, a conforming triangular mesh based representation scheme is implemented to represented DAS in 3D. Second, a curvature guided surface resampling scheme is implemented to reduce the computational cost and to capture high curvature regions. Finally, a modified storage efficient conjugate gradient technique is implemented to update the measured boundary. Experimental evaluations to validate DAS's claim have been performed using several natural and synthetic volumetric images. DAS is demonstrated to be robust to noise, is able to capture high curvature surfaces and have a higher convergence rate compared to parametric methods.

Finally, based on the experimental evaluation of this thesis, a qualitative performance rating table summarizing the capabilities of several parametric and non-parametric methods is shown in Table 7.1.

7.2 Research Contributions

The key contribution of the thesis is the development of a decoupled deformable model (DDM) to rapidly identify the boundary of a 2D/3D object in the presence of outliers. DDM separates the measurements from the prior to solve the conventional active contour or active surface energy minimization problem (MAP, pp. 27) efficiently. By separating the measurements from the prior, DDM first finds an approximate measured boundary

Table 7.1: Summary of qualitative performance of various methods based on this thesis's experimental evaluations. The " * * * * * " is the highest rating. N/A stands for not applicable.

	PAC					NPAC
	DDM	GVF	VFC	PIG	TS	ACWE/LSWR
Initialization insensitivity	* * **	* * *	* * **	* * * * *	*	* * * * *
Robustness to outliers	* * * * *	* * *	* * **	* * **	*	**
Convergence to high curvature	* * **	**	* * *	* * *	*	* * * * *
Scalable to higher dimensions	* * **	* * **	* * **	* * **	* * **	* * * * *
Parameter insensitivity	* * **	**	* * *	* * *	*	* * **
Computational efficiency	* * * * *	**	* * **	* * **	*	*
Identification of broken edges	* * **	* * **	* * **	* * **	* * **	N/A
Handling of multiple topology	N/A	N/A	N/A	* * *	N/A	* * * * *
Handling of embedded topology	N/A	N/A	N/A	**	N/A	N/A

using a ML estimator and then uses a linear Bayesian estimator to incorporate a non-stationary prior into the measured boundary. The non-stationary prior is generated by following a curvature guided sampling method to capture the high curvature regions. The ML estimation step of DDM facilitates the use of a large non-local search space (Chapter 3). As a result, DDM finds an approximate solution from a non-local solution space, making it insensitive to local minima. Unlike conventional methods, DDM integrates a noise model and a non-stationary prior into the conventional traditional deformable model energy functional.

For practical application, this thesis implements the following novel concepts to develop i) decoupled active contour(DAC) [24], ii) fast decoupled active contour (FDAC) [118] and iii) decoupled active surface [119].

1. Formulation of a novel approach to find the measured boundary (an approximate boundary) using hidden Markov Model and Viterbi search.

2. Formulation of an iterative quasi-random search as a scalable and efficient approach to find the measured boundary of a 2D/3D object.

3. Formulation of a curvature guided importance sampling (2D/3D) technique to capture high curvature regions.

4. Formulation of a Bayesian estimator to estimate an updated boundary by fusing the non-stationary prior and the measurement.

5. Formulation of a modified conjugate gradient technique to efficiently solve the Bayesian estimation problem.

In summary, DDM provides a unified framework to integrates historical knowledge (prior) with the measurement to robustly identify the boundary of a 2D/3D object.

7.3 Future Direction

The DDM is an efficient and scalable model, and therefore this DDM model can be extended in future research for multiple boundary detection. DDM is a parametric model and parametric models are not able to handle changes in topology naturally. Therefore, to find multiple objects parametric models use manual multiple initial solutions. An ACID grid based parametric deformable model approach is proposed [120] to identify the boundary of multiple objects. Recently, a Poisson inverse gradient [54] based active contour and active surface is proposed to locate multiple objects. These initialization techniques can be incorporated into DDM to identify multiple boundaries.

The suitability of multi-scale, multi-stage and multi-resolution techniques to further reduce the computational burden of DDM will be studied. The DDM model will be applied to reconstruct 3D surfaces from scattered point clouds (collected using 3D laser scanner) for generating CAD/CAM models of objects. The DDM will be implemented in C/C++ (complied language).

Appendix A

Bayesian Linear Estimator

As discussed in Section 4.3, each element of $\mathbf{v}$ is, itself, an n-vector coordinate $v_j = (x_j, y_j)$; to talk about the estimation of $\mathbf{v}$ is mathematically ambiguous. Therefore, the estimation of the components $\mathbf{x}, \mathbf{y}$ will be explicitly derived (adopted from [121]). Let $\mathbf{v}^e = (\mathbf{x}^e, \mathbf{y}^e)$ be the *true* and unknown discretized deformable model nodes that need to be estimated, considering $\mathbf{v}^e$ to be a random vector with the components of $\mathbf{v}^e$ satisfying a normally distributed prior model Λ with mean $\boldsymbol{\mu}$:

$$\mathbf{x}^e \sim N(\boldsymbol{\mu_x}, \Lambda), \tag{A.1}$$

and

$$\mathbf{y}^e \sim N(\boldsymbol{\mu_y}, \Lambda) \tag{A.2}$$

Assuming a linear relationship between the measured and true states, the extracted and resampled measured boundary $\mathbf{v}^m$ representing the measurements of $\mathbf{v}^e$ can be expressed as:

$$\mathbf{x}^m = C_x \mathbf{x}^e + \boldsymbol{\nu}_x, \tag{A.3}$$

and

$$\mathbf{y}^m = C_y \mathbf{y}^e + \boldsymbol{\nu}_y, \tag{A.4}$$

where $C_x = C_y = I$ are identity matrices, since each visual entity is measured, and where $\boldsymbol{\nu} = [\boldsymbol{\nu}_x, \boldsymbol{\nu}_y]$ is the measurement noise, itself having statistics

$$\boldsymbol{\nu}_x, \boldsymbol{\nu}_y, \sim N(0, R). \tag{A.5}$$

For simplicity, the Bayesian estimation of one component (x) of v for scalar case is derived:

$$\hat{x}_B = \arg_{\hat{x}} \min\{E[(\hat{x}-x)^2|x^m]\} \tag{A.6}$$

$$= \arg_{\hat{x}} \min \int (\hat{x}-x)^2 P(x|x^m)dx. \tag{A.7}$$

To minimize the (A.7) the derivative is set to zero:

$$\frac{d\hat{x}_B}{d\hat{x}} = \frac{\partial}{\partial \hat{x}} \int (\hat{x}-x)^2 P(x|x^m)dx = 0 \tag{A.8}$$

$$= \int 2(\hat{x}-x)P(x|x^m)dx = 0 \tag{A.9}$$

Solving (A.9), we get

$$\underbrace{\int \hat{x}P(x|x^m)dx}_{\hat{x}} = \underbrace{\int xP(x|x^m)dx}_{E(x|x^m)}. \tag{A.10}$$

That is, the sub-optimal estimator is just the conditional mean of x, given the remeasurements. The result for the vector case is identical:

$$\hat{\mathbf{x}} = E[\mathbf{x}|\mathbf{x}^m] \tag{A.11}$$

To solve (A.11) for large estimation problems is difficult. Therefore a simple linear approach that has a closed form solution is used.

Given a linear estimator, $\hat{\mathbf{x}}$ can be written:

$$\hat{\mathbf{x}} = A\mathbf{x}^m + \boldsymbol{\theta}; \tag{A.12}$$

To derive A and $\boldsymbol{\theta}$ the following two criteria are asserted: 1) Unbiasedness and 2) Orthogonality.

Unbiasedness: The estimator is unbiased

$$E[\hat{\mathbf{x}}-\mathbf{x}] = 0 \Longrightarrow E[A\mathbf{x}^m + \boldsymbol{\alpha} - \mathbf{x}] = 0 \tag{A.13}$$

$$\Longrightarrow A\boldsymbol{\mu}_{x^m} + \boldsymbol{\theta} - \boldsymbol{\mu}_x = 0 \Longrightarrow \boldsymbol{\theta} = \boldsymbol{\mu}_x - A\boldsymbol{\mu}_{x^m} \tag{A.14}$$

Orthogonality: The error in $\hat{\mathbf{x}}$ must be un-correlated (orthogonal) with any linear function of data:

$$E[(\hat{\mathbf{x}} - \mathbf{x})(F\mathbf{x}^m + \boldsymbol{\ell})^T] = 0 \qquad \forall F, \ell \tag{A.15}$$

$$\Longrightarrow E[(\hat{\mathbf{x}} - \mathbf{x})]E[(F\mathbf{x}^m + \boldsymbol{\ell})^T] = 0 \tag{A.16}$$

$$\Longrightarrow E[(A\mathbf{x}^m + \boldsymbol{\mu}_x - A\boldsymbol{\mu}_{x^m} - \mathbf{x})(F\mathbf{x}^m + \boldsymbol{\ell})^T] = 0 \tag{A.17}$$

After simplifying (A.17):

$$(A\Lambda_{x^m} - \Lambda_{xx^m})F^T = 0 \Longrightarrow A = \Lambda_{xx^m}\Lambda_{x^m}^{-1} \tag{A.18}$$

where Λ denotes a co-variance or cross-co-variance matrix, $\Lambda_{x^m} = \mathrm{cov}(\mathbf{x}^m)$, $\Lambda_{xx^m} = E[(\mathbf{x} - \boldsymbol{\mu}_x)(\mathbf{x}^m - \boldsymbol{\mu}_{x^m})^T]$. Incorporating the xalue of $\boldsymbol{\theta}$ and A, the Bayesian linear least square estimator becomes:

$$\hat{\mathbf{x}} = Ax^m + \boldsymbol{\theta} = \boldsymbol{\mu}_x + \Lambda_{xx^m}\Lambda_{x^m}^{-1}(\mathbf{x}^m - \boldsymbol{\mu}_{xm}) \tag{A.19}$$

Let $\boldsymbol{\mu}_x = \boldsymbol{\mu}$ and $\Lambda_x = \Lambda$, therefore $\boldsymbol{\mu}_{x^m}$, Λ_{xx^m} and Λ_{x^m} are as shown in (A.22), (A.23), (A.20)

$$\boldsymbol{\mu}_{x^m} = E[(C\mathbf{x} + \boldsymbol{\nu})] = C\boldsymbol{\mu}_x + 0 = C\boldsymbol{\mu} \tag{A.20}$$

$$\Lambda_{xx^m} = E[(\mathbf{x} - \boldsymbol{\mu}_x)(\mathbf{x}^m - \boldsymbol{\mu}_{x^m})^T] \tag{A.21}$$

$$= E[(\mathbf{x} - \boldsymbol{\mu}_x)(C\mathbf{x} + \boldsymbol{\nu} - C\boldsymbol{\mu}_{x^m})^T] = \Lambda x C^T = \Lambda C^T \tag{A.22}$$

$$\Lambda_{x^m} = \mathrm{cov}(\mathbf{x}^m - \boldsymbol{\mu}_x^m) = \mathrm{cov}(C(\mathbf{x} - \boldsymbol{\mu} + \boldsymbol{\nu}) = C\Lambda C^T + R \tag{A.23}$$

Inserting the values of $\boldsymbol{\mu}_x, \boldsymbol{\mu}_{x^m}, \Lambda_{xx^m}$ and Λ_{x^m} in (A.19) and simplifying:

$$\hat{\mathbf{x}} = \boldsymbol{\mu}_x + (\Lambda C^T)(C\Lambda C^T + R)^{-1}(\mathbf{x}^m - C\boldsymbol{\mu}_x) \tag{A.24}$$

Using ABCD lemma (A.12) can be rewritten as:

$$\hat{\mathbf{x}} = \boldsymbol{\mu}_x + (C^T R^{-1} C + \Lambda^{-1})^{-1} C^T R^{-1}(\mathbf{x}^m - C\boldsymbol{\mu}_x) \tag{A.25}$$

References

[1] A.K. Mishra, A. Wong, W. Zhang, D.A. Clausi, and P.W. Fieguth. Improved interactive medical image segmentation using enhanced intelligent scissors (EIS). In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 3083–6, Vancouver, Canada, August 2008. 1, 22

[2] N. Ray, S.T. Acton, and K. Ley. Tracking leukocytes in vivo with shape and size constrained active contours. *IEEE Transactions on Medical Imaging*, 21(10):1222–1235, October 2002. 1, 70

[3] N. Ray and S.T. Acton. Motion gradient vector flow: an external force for tracking rolling leukocytes with shape and size constrained active contours. *IEEE Transactions on Medical Imaging*, 23(12):1466–1478, December 2004. 1

[4] G. Dong, N. Ray, and S.T. Acton. Intravital leukocyte detection using the gradient inverse coefficient of variation. *IEEE Transactions on Medical Imaging*, 24(7):910–924, July 2005. 1, 70

[5] S. Sclaroff and L. Liu. Deformable shape detection and description via model-based region grouping. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(5):475–489, 2001. 1

[6] N. Alajlan, M.S. Kamel, and G.H. Freeman. Geometry-based image retrieval in binary image databases. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(6):1003–1013, 2008. 1

[7] M. Mata, J. Armingol, J. Fernández, and A. De. Object learning and detection using evolutionary deformable models for mobile robot navigation. *Robotica*, 26(1):99–107, 2008. 1

[8] Y. Li, J. Sun, C.K. Tang, and H.Y. Shum. Lazy snapping. *ACM Transactions on Graphics*, 23(3):303–308, 2004. 1, 22

[9] N.M. Eric and A.B. William. Intelligent scissors for image composition. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 191–198, New York, NY, USA, 1995. ACM. 1, 22, 23, 52, 60

[10] MAP3D: Interactive scientific visualization tool for bioengineering data. Scientific Computing and Imaging Institute (SCI). 1

[11] R. Bowden, T.A. Mitchell, and M. Sahardi. Real-time dynamic deformable meshes for volumetric segmentation and visualisation. In *Bristish Machine Vision Conference*, pages 310–319, 1997. 1

[12] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. Segmentation and recognition using structure from motion point clouds. In *European Conference on Computer Vision*, pages 44–57, Berlin, Heidelberg, 2008. Springer-Verlag. 1

[13] A.P. Pentland. Automatic extraction of deformable part models. *International Journal of Computer Vision*, 4(2):107–126, March 1990. 1

[14] A.P. Pentland and S. Sclaroff. Closed-form solutions for physically based shape modeling and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):715–729, July 1991. 1

[15] M. Bertalmío, G. Sapiro, and G. Randall. Morphing active contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):733–737, 2000. 1

[16] M. Bertalmio, G. Sapiro, and G. Randall. Morphing active contours: a geometric approach to topology-independent image segmentation and tracking. In *IEEE International Conference on Image Processing*, pages III: 318–322, 1998. 1

[17] C.R. Shelton. Morphable surface models. *International Journal of Computer Vision*, 38(1):75–91, June 2000. 1, 8

[18] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2247–2253, December 2007. 2, 120

[19] A. Wong, A. Mishra, J. Yates, P.W. Fieguth, D.A. Clausi, and J.P. Callaghan. Intervertebral disc segmentation and volumetric reconstruction from peripheral quantitative computer tomography imaging. *IEEE Transactions on Biomedical Engineering*, 56(11):2748–2751, 2009. 4

[20] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988. 1, 3, 4, 5, 8, 9, 13, 14, 20, 21, 27, 31, 32, 45, 51, 52, 53, 54, 55, 56, 57, 58, 59, 62, 63, 64, 68

[21] C.Y. Xu and J.L. Prince. Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing*, 7(3):359–369, 1998. 3, 4, 5, 8, 9, 13, 14, 17, 18, 20, 21, 22, 23, 31, 32, 45, 51, 52, 53, 54, 55, 56, 57, 58, 59, 62, 63, 64, 65, 68, 86, 87, 97, 99

[22] B. Li and S. T. Acton. Active contour external force using vector field convolution for image segmentation. *IEEE Transactions on Image Processing*, 16(8):2096–2106, 2007. 3, 4, 5, 9, 13, 14, 17, 19, 20, 21, 22, 23, 32, 85, 86, 87, 88, 89, 90, 94, 97, 99

[23] L.D. Cohen. On active contour models and balloons. *Computer Vision Graphics and Image Understanding*, 53(2):211–218, March 1991. 3, 5, 17, 18, 21, 22, 45, 46, 97

[24] A. Mishra, P.W. Fieguth, and D. A. Clausi. Decoupled active contour (DAC) for boundary detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Preprint, 2010. 3, 5, 32, 88, 89, 90, 125

[25] M. Krueger, P. Delmas, and G. Gimel'Farb. Active contour based segmentation of 3D surfaces. In *European Conference on Computer Vision*, pages 350–363, Berlin, Heidelberg, 2008. Springer-Verlag. 3

[26] J. Mille, R. Boné, and L. D. Cohen. Region-based 2D deformable generalized cylinder for narrow structures segmentation. In *European Conference on Computer Vision*, pages 392–404, Berlin, Heidelberg, 2008. Springer-Verlag. 3

[27] F. Leymarie and M.D. Levine. Tracking deformable objects in the plane using an active contour model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):617–634, 1993. 4, 23, 70

[28] L.D. Cohen and I. Cohen. Finite-element methods for active contour models and balloons for 2-D and 3-D images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1131–1147, 1993. 4, 5, 21, 22

[29] F.Y. Shih and K. Zhang. Locating object contours in complex background using improved snakes. *Computer Vision Graphics and Image Understanding*, 105(2):93–98, 2007. 4, 16, 22, 68

[30] G. Storvik. A Bayesian approach to dynamic contours through stochastic sampling and simulated annealing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(10):976–986, 1994. 4, 26, 27

[31] P. Brigger, J. Hoeg, and M. Unser. B-spline snakes: a flexible tool for parametric contour detection. *IEEE Transactions on Image Processing*, 9(9):1484–1496, 2000. 4

[32] Y.Y. Wong, P.C. Yuen, and C.S. Tong. Segmented snake for contour detection. *Pattern Recognition*, 31(11):1669–1679, November 1998. 4, 21, 40

[33] G. Sundaramoorthi, A.J. Yezzi, Jr., and A. Mennucci. Coarse-to-fine segmentation and tracking using sobolev active contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):851–864, 2008. 4

[34] G.G. Slabaugh and G. Unal. Active polyhedron: Surface evolution theory applied to deformable meshes. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages II: 84–91, 2005. 4, 8, 23, 110

[35] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–97, 1997. 4, 15, 24, 45, 51, 97

[36] X. Bresson, S. Esedoglu, P. Vandergheynst, J. Thiran, and S. Osher. Fast global minimization of the active contour/snake model. *Journal of Mathematical Imaging and Vision*, 28(2):151–167, 2007. 4, 16, 20, 24, 27, 51, 64

[37] 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. 4, 8, 15, 24, 32, 51, 52, 60, 97

[38] V. Caselles, F. Catte, T. Coll, and F. Dibos. A geometric model for active contours in image processing. *Numerische Mathematik*, 66:1–31, 1993. 4, 15

[39] T.F. Chan, B.Y. Sandberg, and L.A. Vese. Active contours without edges for vector-valued images. *Journal of Visual Communication and Image Representation*, 11(2):130–141, 2000. 4, 16, 24, 32, 97

[40] T.F. Chan and L.A. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, 2001. 4, 16, 31, 32, 51, 52, 53, 54, 55, 56, 57, 58, 62, 63, 64, 68, 97

[41] S. Zhu and A. Yuille. Region competition: Unifying snakes, region growing, and Bayes/MDL for multiband image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):884–900, 1996. 4

[42] E. Debreuve, M. Barlaud, G. Aubert, I. Laurette, and J. Darcourt. Space-time segmentation using level set active contours applied to myocardial gated spect. *IEEE Transactions on Medical Imaging*, 20(7):643–659, July 2001. 4

[43] A.C. Jalba, M.H.F. Wilkinson, and J.B.T.M. Roerdink. CPM: A deformable model for shape recovery and segmentation based on charged particles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10):1320–1335, 2004. 4, 8, 16

[44] F. Eric, F. Gibou, and R. Fedkiw. A fast hybrid k-means level set algorithm for segmentation. Technical report, 4th Annual Hawaii International Conference on Statistics and Mathematics, Honolulu Hawaii, USA, 2002. 4, 16

[45] R. Malladi, J.A. Sethian, and B.C. Vemuri. A fast level set based algorithm for topology-independent shape modeling. *Journal of Mathematical Imaging and Vision*, 6(2-3):269–289, 1996. 4

[46] K. Siddiqi, S. Zucker, Y. BérubéLauzière, and A. Tannenbaum. Area and length minimizing flows for shape segmentation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 621–627, Washington, DC, USA, 1997. IEEE Computer Society. 4, 15

[47] C. Tejos, P. Irarrazaval, and A. Cardenas Blanco. Simplex mesh diffusion snakes: Integrating 2D and 3D deformable models and statistical shape knowledge in a variational framework. *International Journal of Computer Vision*, 85(1), 2009. 4

[48] Shawn Lankton. Sparse field methods - technical report. Technical report, Georgia Institute of Technology, April 2009. 4

[49] E. Sifakis, I. Grinias, and G. Tziritas. Video segmentation using fast marching and region growing algorithms. *EURASIP Journal on Applied Signal Processing*, 2002(4):379–388, 2002. 4

[50] M. Wang, J. Evans, L. Hassebrook, and C. Knapp. A multistage, optimal active contour model. *IEEE Transactions on Image Processing*, 5(11):1586–1591, November 1996. 4

[51] J.A. Schnabel and S.R. Arridge. Multi-scale active shape description. In *ScaleSpace*, page xx, 1997. 4

[52] A.A. Amini, T.E. Weymouth, and R.C. Jain. Using dynamic programming for solving variational problems in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(9):855–867, 1990. 5, 22, 23

[53] C.M. Li, C.Y. Xu, C. Gui, and M.D. Fox. Level set evolution without re initialization a new variational formulation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages I430–436, San Diego, CA, USA, 2005. ACM. 5, 15, 97

[54] L. Bing and S.T. Scott. Automatic active model initialization via poisson inverse gradient. *IEEE Transactions on Image Processing*, 17(8):1406–1420, 2008. 5, 17, 23, 85, 86, 88, 89, 90, 94, 97, 99, 126

[55] A. Mishra, P.W. Fieguth, and D. A. Clausi. Accurate boundary localization using dynamic programming on snakes. In *IEEE International Conference on Image Processing*, pages 1092–1095, Florida, USA, September 2008. 5, 87

[56] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002. 6

[57] R.R. Kohler. A segmentation system based on thresholding. *Computer Vision Graphics and Image Understanding*, 15(4):319–338, April 1981. 8

[58] Z. Hou, Q.M. Hu, and W.L. Nowinski. On minimum variance thresholding. *Pattern Recognition Letter*, 27(14):1732–1743, October 2006. 8

[59] L. Vincent and P. Soille. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):583–598, June 1991. 8

[60] Q. Yu and D.A. Clausi. IRGS: image segmentation using edge penalties and region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(12):2126–2139, 2008. 8

[61] Q. Yu and D.A. Clausi. SAR sea-ice image analysis based on iterative region growing using semantics. *IEEE Transactions on Geoscience and Remote Sensing*, 45(12):3919–3931, 2007. 8

[62] Lovell B. C. Biswas, S. *Snakes and Active Contours*, pages 187–212. Springer London, Thirteenth edition, December 2007. 8

[63] G. Tsechpenakis and D.N. Metaxas. CoCRF deformable model: A geometric model driven by collaborative conditional random fields. *IEEE Transactions on Image Processing*, 18(10), 2009. 8

[64] X. Xianghua and M. Majid. MAC: Magnetostatic active contour model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(4):632–646, 2008. 8, 16

[65] D.P. Perrin and C.E. Smith. Rethinking classical internal forces for active contour models. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages II:615–620, Kauai Marriott, Hawaii, USA, December 2001. 8

[66] B.C. Vemuri and R. Malladi. Deformable models: Canonical parameters for surface representation and multiple view integration. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 724–725, 1991. 8

[67] Y.F. Wang and J.F. Wang. Surface reconstruction using deformable models with interior and boundary constraints. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(5):572–579, May 1992. 8

[68] S. Sullivan, L. Sandford, and J. Ponce. Using geometric distance fits for 3-D object modeling and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(12):1183–1196, December 1994. 8

[69] S. Sclaroff and J. Isidoro. Active blobs: region-based, deformable appearance models. *Computer Vision and Image Understanding*, 89(2–3):197–225, February 2003. 8

[70] W. Almageed and C.E. Smith. Active deformable models using density estimation. *International Journal of Image and Graphics*, 4(3):343–361, July 2004. 8

[71] S. Osher and J.A. Sethian. Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79(1):12–49, 1988. 15, 16

[72] L.D. Cohen and R. Kimmel. Global minimum for active contour models: A minimal path approach. *International Journal of Computer Vision*, 24(1):57–78, 1997. 16

[73] D. Mumford and J. Shah. Boundary detection by minimizing functionals. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 22–26, San Francisco, CA, June 1985. 16, 58

[74] G. Sundaramoorthi, A.J. Yezzi, Jr., and A.C. Mennucci. Sobolev active contours. *International Journal of Computer Vision*, 73(3):345–366, 2007. 16

[75] J.C. Nascimento and J.S. Marques. Adaptive snakes using the EM algorithm. *IEEE Transactions on Image Processing*, 14(11):1678–1686, November 2005. 20, 31

[76] A. Blake and Isard M. *Acitive Contours*. Springer, New York, 1998. 20

[77] Timothy F. C., Christopher J. T., David H. C, and Jim Graham. Active shape models-their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995. 20

[78] A. Blake, R.M. Curwen, and A. Zisserman. A framework for spatiotemporal control in the tracking of visual contours. *International Journal of Computer Vision*, 11(2):127–145, October 1993. 20

[79] J.C. Nascimento and J.S. Marques. Robust shape tracking in the presence of cluttered background. In *IEEE International Conference on Image Processing*, pages Vol III: 82–85, 2000. 20, 31

[80] J.C. Nascimento and J.S. Marques. Robust shape tracking with multiple models in ultrasound images. *IEEE Transactions on Image Processing*, 17(3):392–406, March 2008. 20

[81] Y. Bar-Shalom and Fortmann T. *Tracking and Data Association.* Academic, New York, 1998. 20

[82] G. Xu, E. Segawa, and S. Tsuji. Robust active contours with insensitive parameters. *Pattern Recognition*, 27(7):879–884, July 1994. 21

[83] G. Xu, E. Segawa, and S. Tsuji. A robust active contour model with insensitive parameters. In *International Conference on Computer Vision*, pages 562–566, 1993. 21

[84] A.A. Amini, S. Tehrani, and T.E. Weymouth. Using dynamic programming for minimizing the energy of active contours in the presence of hard constraints. In *International Conference on Computer Vision*, pages 95–99, Bombay, India, January 1988. 22

[85] A. Amini, T. Weymouth, and R. Jain. Using dynamic programming for solving variational problems in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(9):855–867, 1990. 22

[86] S. Chandran and A.K. Potty. Energy minimization of contours using boundary conditions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):546–549, 1998. 22

[87] G. Storvik. A Bayesian approach to dynamic contours through stochastic sampling and simulated annealing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(10):976–986, 1994. 22, 23

[88] D. Williams and M. Shah. A fast algorithm for active contours and curvature estimation. *Computer Vision Graphics and Image Understanding*, 55(1):14–26, 1992. 22, 52, 60

[89] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23(3):309–314, 2004. 22, 23

[90] H. Chang, Q. Yang, and C. Pan. An iterative Bayesian approach for digital matting. In *IEEE International Conference on Pattern Recognition*, pages II: 122–125, 2006. 22

[91] D. Molloy and P.F. Whelan. Active-meshes. *Pattern Recognition Letter*, 21(12):1071–1080, November 2000. 23

[92] A.J. Bulpitt and N.D. Efford. An efficient 3D deformable model with a self-optimizing mesh. *Image and Vision Computing*, 14(8):573–580, August 1996. 23

[93] J.P. Pons and J.D. Boissonnat. Delaunay deformable models: Topology-adaptive meshes based on the restricted delaunay triangulation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007. 23

[94] D. Terzopoulos and R. Szeliski. *Tracking with Kalman snakes*, pages 3–20. MIT Press, Cambridge, MA, USA, 1993. 26

[95] A.K. Jain, Y. Zhong, and M.P. Dubuisson Jolly. Deformable template models: A review. *Signal Processing*, 71(2):109–129, December 1998. 26, 27

[96] J. Montagnat, H. Delingette, and N. Ayache. A review of deformable surfaces: topology, geometry and deformation. *Image and Vision Computing*, 19(14):1023–1040, December 2001. 26, 27, 97, 99

[97] S. Alpert, M. Galun, T. Basri, and A. Brandt. Image segmentation by probabilistic bottom-up aggregation and cue integration. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–8, Minneapolis, MN, USA, June 2007. 31, 51, 53, 54, 68

[98] A.J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, IT-13(2-3):260–269, 1967. 32, 33, 38, 40

[99] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. 33, 38

[100] H. Li and J. Yezzi. Local or global minima: Flexible dual-front active contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):1–14, 2007. 36

[101] D.A. Forsyth and J. Ponce. *Computer Vision A Modern Approach*. Prentice Hall, Upper Saddle River, New Jersey 07458, 2003. 41

[102] A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill Companies, 3rd edition, 1991. 44

[103] E.L. Lehmann and G. Casella. *Theory of Point Estimation*. Springer, 2nd ed, 1998. 44

[104] N. Vaswani and R. Chellappa. Principal components null space analysis for image and video classification. *IEEE Transactions on Image Processing*, 15(7):1816–1830, 2006. 45

[105] C. Xu and J. Prince. Johns Hopkins University's Image Analysis and Communications lab's GVF software, 1999. http://iacl.ece.jhu.edu/projects/gvf/. 52, 55

[106] S. Lankton. Georgia Institute of Technology, 2007. http://www.shawnlankton.com/2007/05/active-contours/. 52

[107] The USC-SIPI Image Database. University of southeren california. http://sipi.usc.edu/database/. 54

[108] R.K.S. Kwan, A.C. Evans, and G.B. Pike. MRI simulation-based evaluation of image-processing and classification methods. *IEEE Transactions on Medical Imaging*, 18(11):1085–1097, 1999. 54, 68

[109] K. Bay. Starfish- they are huge, 2007. http://kahunabay.wordpress.com/2007/09/07/starfish-they-are-huge/. 55, 68

[110] C.A Davatzikos and J.L. Prince. An active contour model for mapping the cortex. *IEEE Transactions on Medical Imaging*, 14(1):65–80, March 1995. 70

[111] I. Sobol. Uniformly distributed sequences with an additional uniform property. *USSR Computational Mathematics and Mathematical Physics*, 16:236–242, 1977. 77

[112] A. Mishra, A. Wong, D.A. Clausi, and P.W. Fieguth. Quasi-random nonlinear scale space. *Pattern Recognition Letters, In Press, Corrected Proof*, –:–, 2010. 77

[113] B. Li and S.T. Acton. Verginia image and video analysis, vector field convolution, 2009. http://viva.ee.virginia.edu/researchvfc/. 85

[114] Y. Zhang, B. J. Matuszewski, L. Shark, and C. J. Moore. Medical image segmentation using new hybrid level-set method. In *MEDIVIS '08: Proceedings of the 2008 Fifth International Conference BioMedical Visualization: Information Visualization in Medical and Biomedical Informatics*, pages 71–76, Washington, DC, USA, 2008. IEEE Computer Society. 97, 114, 115, 116, 117, 118, 121

[115] K. Varanasi, A. Zaharescu, E. Boyer, and R. Horaud. Temporal surface tracking using mesh evolution. In *European Conference on Computer Vision*, pages II: 30–43, 2008. 110

[116] Z. Yan. University of central lancashire, 2009. http://www.mathworks.com/matlabcentral/fileexchange/authors/65850. 114

[117] PETS. Tenth ieee international workshop on performance evaluation of tracking and surveillance, 2007. http://pets2007.net/. 118, 119

[118] A. Mishra, P.W. Fieguth, and D.A. Clausi. Fast decoupled active contour (FDAC) for boundary detection. *IEEE Transactions on Image Processing*, Will be submitted, June, 2010. 125

[119] A. Mishra, P.W. Fieguth, and D. A. Clausi. Decoupled active suraface for volumetric image segmentation. In *Canadian Conference on Robotics and Computer Vision*, Ottawa, Canada, May 2010. 125

[120] T. McInerney and D. Terzopoulos. T-snakes: Topology adaptive snakes. *Medical Image Analysis*, 4:73–91, 2000. 126

[121] P.W Fieguth. Course note: Multi-dimensional signal proceesing, 2007. Systems Design Engineering, University of Waterloo. 127