University of Alberta

The evolution of snake toward automation for multiple blob-object segmentation

by

Baidya Nath Saha

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Computing Science

Baidya Nath Saha
Fall 2011
Edmonton, Alberta

Dedication

I dedicate my dissertation work to my family and my teacher Professor Dipti Prasad Mukherjee, I am greatly indebted to all of you.

To my father **Nirmal Kumar Saha**, I am forever grateful for providing endless support to me for pursuing higher education even in the most adverse circumstances.

To my mother **Kanchan Lata Saha**, thank you for your care, love and living with the anxiety associated with me staying away from home.

To my dear and sweet wife **Jhuma Saha**, so grateful for your presence in my life, for your encouragement, patience in being a wife, a mother, a perfect homemaker, and a student all at the same time.

To my little son, **Avighna Saha**, I express deep gratitude to my little one. When you were smiling for me I was given more hope and energy to complete the dissertation.

To my mother-in-law **Shanti Saha**, thank you so much for your invaluable support, encouragement, love and care for our newborn baby Avighna that made it possible for me to work during some of the most difficult phases of this dissertation.

To my teacher Professor **Dipti Prasad Mukherjee**, thank you for everything that you have done towards supporting me in my admission to the Ph.D. program. I am so lucky to find an ideal teacher like you in my life.

Thank you all so much for being a part of my life.

Abstract

For the last two decades “active contour” or “snake” has been effective as an interactive image segmentation tool in a wide range of applications, especially for blob-object delineation. In the interactive snake segmentation process, a user draws a rough object outline; next, a cost function is optimized to drive the user-drawn contour a.k.a. snake to delineate the desired object boundary. Although successful as an interactive segmentation tool, snake exhibits poor performances in various noteworthy image segmentation applications that require complete automation. Examples include oil sand particle delineation, biological cell segmentation and so on.

This thesis presents a novel, completely automated snake/active contour algorithm for multiple blob-object delineation. The algorithm consists of three sequential steps: (a) snake initialization: where we apply a Probabilistic Quad Tree (PQT) based approximate segmentation technique on an image to find the regions of interest (ROI) where the probability of having objects is very high and place seeds uniformly within the ROIs; (b) snake evolution: where we evolve one novel interleave directional gradient vector flow (IDGVF) snake from each seed; (c) snake validation: where we classify the snakes into object and non-object classes using a novel adaptive regularized boosting (ARboost). Existing efforts towards snake automation have concentrated only on the succession of initialization and evolution steps and have practically overlooked the snake validation step. Here, we emphasize that we cannot skip the validation step, even though the initialization and evolution have performed well. Our proposed novel

validation step, executed after complete convergence of a snake contour from a given initialization, classifies the evolved contour into desired object and non-object classes.

ARboost employs a novel loss function for boosting that enables to classify snakes more accurately into object and non-object classes than other variants of boosting. PQT generates substantially fewer seed points and is therefore more efficient than other initialization methods without degrading the segmentation performance. We have demonstrated that IDGVF is more robust to initialization and it possesses a broader capture range than other variants of GVF snakes.

The proposed automated snake algorithm has been successfully applied to two real data sets: oil sand ore images that have relevance in the oil sand mining industry and leukocyte images that are significant in biomedical engineering.

Acknowledgements

This thesis would not be complete without expressing my deep gratitude to a number of people who generously offered friendship, inspiration, advice, and encouragement throughout my Ph.D. First of all, I express my sincere gratitude to my teacher and thesis supervisor, Dr. Nilanjan Ray, with whom it has been an honor and a pleasure to work. I am indebted to him for a number of things: for sharing his knowledge, for his friendly disposition, for the opportunities he provided me, for giving me complete freedom, for his patience, for his accessibility, for his invaluable suggestions and many more reasons. I would like to modestly mention here that he has had a profound and everlasting influence on my career and life.

Many thanks to Dr. Hong Zhang for sharing his ideas, experience, helpful suggestions and constructive criticism that have helped me in reaching this goal. Special thanks to Dr. Russell Greiner for his critical review of different aspects of this research and helping me to improve the thesis. Dr. Hong Zhang and Dr. Russell Greiner have not only been very insightful supervisory committee members, but also a source of inspiration in my research. The entire examination committee deserves special thanks for the very positive attitude and encouragement in this research.

I acknowledge the support received from the present as well as the former CIMS (Centre for Intelligent Mining System) team members towards the completion of this work. The Computing Science Department deserves thanks for providing me with state-of-the-art research facilities. I greatly acknowledge NSERC discovery grant, Center for Intelligent Mining Systems (CIMS), Mitacs internship award and Department of Computing Science, University of Alberta to provide financial support for this research work.

During my years in Edmonton, I have been lucky to have a lot of great friends and colleagues. Thank you to all the friends and colleagues who have made my student life in Edmonton a whole lot more enjoyable. Mr. Partha Roy, Abhishek Srivastava and Mrs. Rupa Ray deserve special mention here; without receiving their greatest support in numerous personal occasions, it would have been impossible to reach this goal.

I am also thankful to Adrija Sarkar, Mrs. Ananya Sarkar and all of my other family members and friends, to whom I would like to express my apologies for being unable to mention all of their names here owing to space restriction.

Last but not the least, I feel proud to specially acknowledge and extend my heartfelt gratitude to Apurbalal Senapati, my best friend and Subhasis Maiti of ‘Breakthrough Science Society’, Kolkata who have been strong motivating factors in my life. They have had enormous influence in changing my outlook towards life in a positive manner. I will forever be indebted to them. I wish you all the best in your life.

Contents

List of Figures

List of Tables

GLOSSARY

QT	Probabilistic Quad Tree
PQT	Probabilistic Quad Tree
GVF	Gradient Vector Flow
GGVF	Generalized Gradient Vector Flow
EGGVF	Enhanced Generalized Gradient Vector Flow
IDGVF	Interleave Directional Gradient Vector Flow
ARboost	Adaptive Regularized Boosting
FFS	Force Field Segmentation
VFC	Vector Field Convolution
PIG	Poisson Inverse Gradient
COD	Center of Divergence
CP	Critical Point
PCA	Principal Component Analysis
ROIs	Regions of Interest
UI	Uniform Initialization
GICOV	Gradient inverse Coefficient of Variation
ROC	Receiver Operating Characteristic
PFOM	Pratt’s Figure of Merit
PSV	Primary Separation Vessel
BDF	Bayesian Discriminative Feature
exp(.)	The exponential function
div	The divergence operator

diag	The diagonal operator
$I(x, y)$	Represents an image, a continuous function mapping a Point (x, y) to a real number
∇	The gradient operation
$\lvert \nabla I \rvert$	The absolute value of the gradient magnitude of $I\,(x, y)$
$\Re^{N}$	Real Euclidean space of dimension N
$\Re^{N\times N}$	Real Euclidean space of dimension N-by-N
∂C	Boundary of the region C
$\mathbf{n}_{\partial C}$	Unit outward normal to the boundary of region C
$D\backslash C$	The set difference between region D and region C
Δ	The laplacian operator
$*$	The convolution operator
f_x	Abbreviation for $\frac{\partial f}{\partial x}$
(u, v)	Snake external force field
I_N	n-by-n identity matrix

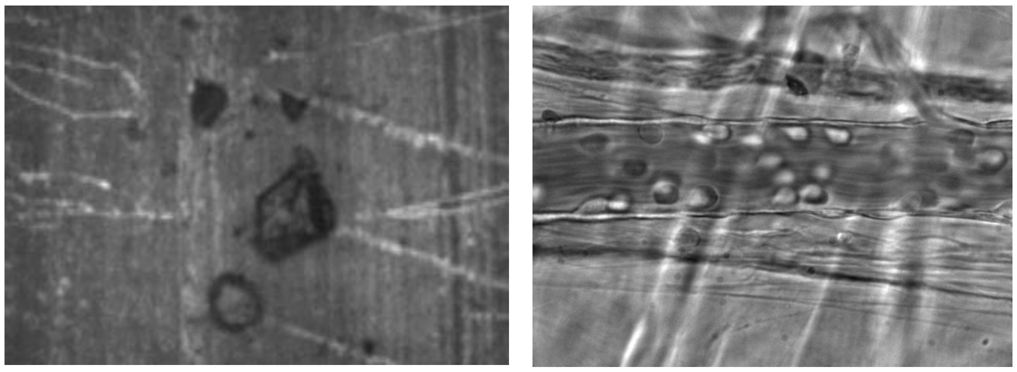
Chapter 1

1. Introduction

The "Active Contour" or "Snake" is a very popular interactive image segmentation techniques used in a wide range of applications [21], especially for blob-object segmentation since its introduction by *Kass et al*. in 1987 [21]. The introductory paper "Snakes: Active Contour Models" originated by Kass, Witkin and Terzopoulos [21] received more than *ten thousand* citations till date according to Google Scholar [50]. The interactive active contour or snake segmentation technique consists of two sequential steps: (a) snake initialization: where a user manually places a rough object boundary near the actual object edges; and (b) snake evolution: where, an energy functional is minimized that provides necessary force to move the user-drawn contour aka snake towards desired features, usually edges. The energy functional is designed in such a way so that energy becomes minimum along the actual object edges and the snake contour eventually locks onto actual object boundaries. Thus, the snake contour delineates the region (blob) surrounding the desired feature (edge). Snake energy functional consists of two terms: (a) internal energy– it helps to make the snake contour smooth and continuous; and (b) external energy– it provides the necessary force to move snake contour towards desired features such as line or edges.

Traditional low-level vision tasks such as edge detection [4], [27] techniques developed prior to snake have been treated as autonomous bottom-up processes. These traditional edge detection techniques consist of a number of sequential steps such as, noise removal, local maxima suppression and so on. Lack of the opportunity of correcting the mistakes made up at different steps of these traditional edge detection techniques propagates at the end. It is very difficult also for incorporating high level information into low-level vision tasks. As a result, these traditional edge detection techniques exhibit poor performance. One of the advantages of snakes over traditional edge based segmentation techniques is that snake allows to combine high level information (priori knowledge of object shape and inherent smoothness, usually formulated as internal energy of the snake) and low level information (features such as edges, usually formulated as external energy) into a single variational energy functional that helps snake to be more insensitive to noise, gaps and other irregularities in object boundaries than traditional edge-based techniques. Variational framework also reduces the number of parameters of the algorithm than the traditional edge based segmentation techniques [32] .

In the area of computer vision, multiple blob-object delineation aims at seeking solid connected components (blobs) in an image. Snake delineates a blob-object boundary by considering it as a single, connected structure [28]. For multiple blob-object delineation, blob boundaries are delineated by distinct snake contours. Snake contour does not split or merge due to its inherent topological inflexibility, which in turn helps to delineate two touching blob-objects using two separate snake contours.


(a) (b)
Figure 1.1: (a) oil sand image. (b) Leukocyte image.

In this thesis, we have presented two emerging applications that require *automated*, multiple blob-object segmentation: oil sand particle delineation from oil sand images [46] and leukocyte detection [10] from intravital microscopy images. Both oil sand particles and leukocytes are blob-objects. Oil sand particles considered in this study are relatively darker and leukocytes are relatively brighter from the rest of the images as shown in Figure 1.1. Oil sand mining industry developed a novel crushing less technology that produce no rejects of oil sand particles and is therefore contribute to better recovery rates since no oil sand particle is rejected that could produce oil [25]. One additional advantage of this technology is that there is no expensive vibrating screen required to reject crushed but bigger oil sand particles coming out of the crusher from rest of the smaller particles. Because, smaller particles coming out from the crusher are passed over vibrating screen in the traditional oil sand extraction process. Oversize particles are rejected and undersize particles are passed through vibrating screen and then are sent to the hydro transport plant for further processing towards bitumen extraction. Oil sand mining industry is interested to validate the design of the novel crusher used in this new technology. To automatically validate the design of this crusher, images of the crushed particles coming out of the crusher are captured by a video camera. Then oil sand images are segmented. Subsequently, the size of each of the crushed oil sand particle is measured and is investigated whether the size of each crushed particle exceeds an expected size.

In another significant application, leukocytes play an important role in inflammation by transmigrating through endothelium and accumulating at the site of injury. Inflammation is a normal defense mechanism; however, it becomes injurious in inflammatory diseases. Inflammation research entails study of the velocity distribution of leukocytes [36]. Bio-medical research laboratories are conducting experiments on living mice where the velocities of the leukocytes inside the veins of a mouse cremaster muscle are observed in video recordings made though a camera coupled with the intravital microscope. Leukocyte rolling velocity is an indicator of the inflammation process. To obtain the velocities of leukocyte or to initiate tracking of leukocytes, leukocytes are required to segment first. In both of these applications, skilled operators are employed to manually inspect hours of video and to extract the desired regions. Such manual processing is subject to operator errors and biases, extremely time consuming, tedious, has poor reproducibility and above all not cost effective.

In this research we have demonstrated that the “Active contour” or “Snake” can successfully delineate oil sand particles and leukocytes [46], [37]. Literature also reveals that snake has been a successful interactive image segmentation tool in a wide range of applications for the last two decades [21]. However, snake demonstrates poor performance in many applications that require complete automation [22]. This has motivated us for automating snakes to serve the purpose of automation where snake serves well as an interactive image segmentation tool.

1.1 Problem Statement

Literature survey shows that interactive segmentation techniques, such as, active contour or snake [21] is an effective alternative to other traditional automatic segmentations in many applications. While automatic segmentation can be very challenging, a small amount of user input can often resolve ambiguous decisions on the part of the algorithm. Although successful as an interactive image segmentation technique, snakes fail to automatically delineate multiple blob-objects. Literature reveals that there are two potential roadblocks for snake automation: (a) poor convergence and (b) automatic initialization [22]. The success of the snake algorithm on delineating accurate object boundary depends on the placement of the initial contour near the object boundary. If this placement is far away from the object boundary, the contour cannot be guaranteed to lock onto the object edges. Depending on the requirement of the application, initial contour can be placed manually or with some automatic means. One difficult roadblock in automating the snake/active contour is the automated initialization. To overcome this difficulty, several automated initialization techniques have been proposed [22]. Literature shows that automatic snake initialization techniques lead to over segmentation and often require post processing in the form of region merging [22]. For segmenting multiple blobs using active contour or snake, ideally, the number of seed points needed should be the same as the number of blobs present in the image and snakes evolved from each seed should be able to capture distinct objects. However, clutter available in the noisy images misguides the automatic seed placement techniques and a number of spurious snakes are generated from the unexpected seeds. Thus, snake/active contour has been shown to be effective as an interactive image segmentation tool to delineate blob-like objects like oil sand particles or leukocytes. We now define the research problem statement as: *Development of a completely automated snake based segmentation algorithm for multiple blob-object delineation*.

1.2 Proposed Solution

In this thesis, we have advocated that for complete automation of snake, we need three sequential and automated steps: (a) snake initialization, where we first find regions of interest (ROIs) and place seeds uniformly within ROIs, (b) snake evolution, where we evolve one snake from each seed, and (c) snake validation, where we classify snakes into object and non-object classes. Significant research has been conducted only on the progression of snake initialization and evolution steps towards snake automation and the validation step has been practically ignored. Here, we emphasize that we cannot disregard the validation step, because snake initialization suffers from over segmentation. Clutter in the noisy images produces many unwanted seeds and snakes evolved from these seeds do not converge at the true object edges. When snakes are fully converged from given initializations, our proposed validation step classifies snakes into desired object and non-object classes using principal component analysis (PCA) or boosting-based classifier that we will demonstrate later. This snake classification algorithm removes unwanted snakes generated due to clutters. Three steps of our proposed snake automation algorithm are as follows.

Snake Initialization: Initialization zones, called regions of interest (ROIs), are identified by fast and approximate segmentation techniques. The probabilities of locating objects within ROIs are higher than the rest of the image. We implement Probabilistic Quad Tree (PQT) based approximate segmentation technique to find ROIs.

Snake Evolution: We throw seeds uniformly within ROIs and allow Interleave Directional Gradient Vector Flow (IDGVF) snakes to evolve from seed points. We have incorporated directional gradient information along with Dirichlet boundary condition into GVF framework that makes the GVF force field anisotropic and facilitates snakes to become more initialization independent. We compute the force field in an interleave fashion that significantly enhances the capture range of IDGVF snake.

Snake Validation: When all snakes converge, each evolved snake contour is sieved through a classifier to verify whether the image area delineated by the snake is indeed the desired object. We have proposed two snake validation techniques using (a) Principal Component Analysis (PCA) and (b) Boosting that are discussed as follows.

a) Snake Validation by PCA: We have employed PCA-based classifier for snake validation. To perform classification using PCA, we first construct a novel rectangular pattern image formed by unfolding an annular band across each converged snake contour. Then each pattern image is projected into an already trained PCA space and PCA reconstruction error is computed. PCA training is performed on pattern images associated with actual objects marked by experts on training images. The snake associated with lower PCA reconstruction error is regarded as object, otherwise it is considered as non-object. Pattern image across the converged snake contour carries intensity transition (dark-to-bright transition or vice-versa) across the object boundaries. This intensity transition seems to be fuzzy for converged snake contour across clutter. This pattern image is effective for discriminating objects from clutter if prominent object edges are present.

b) Snake Validation by boosting: We have enhanced the performance of abovementioned PCA-based snake validation using boosting. In boosting-based snake validation we have exploited the advantages of multiple features over single feature (intensity transitions across actual object edges) used in the PCA for snake classification into object and non-object classes. We propose a novel loss function for boosting that offers optimal weight on the misclassified samples at each stage of boosting iteration. Optimal weight enforces misclassified samples to classify correctly in the subsequent iterations and thus results in faster convergence than the Adaboost algorithm. We have derived a slightly modified Adaboost algorithm based on the proposed loss function. We have carried out experiments on both oil sand and leukocyte images. Results show that the proposed snake validation technique is an indispensable part of snake-based automatic segmentation technique and it augments the performances of existing automatic initialization techniques [13], [54] proposed in literature towards snake automation.

1.3 Contributions

- To overcome the difficulties encountered by existing researches on snake based automatic segmentation proposed in literature, we take a different route for snake automation that has hardly received any attention in snake related research. We first proposed in literature that snake-based automatic segmentation consists of three sequential steps: snake initialization, snake evolution and snake validation [47]. Existing efforts contemplated only on the development of snake initialization and evolution step and practically ignores the validation step. We have demonstrated that over segmentation resulted in existing automated snake segmentation algorithm consisting of step [47].
- We have proposed a probabilistic quad tree (PQT) based approximate segmentation technique [48] for snake initialization that accelerates the snake algorithm for multiple objects detection by generating fewer seeds than its competitors without degrading the performance.

- We have enhanced the performances of the existing GGVF snake evolution significantly by incorporating directional gradient information as well as Dirichlet boundary condition into GGVF energy functional [46]. We note that oil sand particle and leukocyte boundaries are characterized by dark-to-bright and bright-to-dark intensity transitions going from inside to outside. We have accommodated this prior information inside governing partial differential equation that computes the force field. Dirichlet Boundary Condition (DBC) encourages the initial snake to grow, so that if the initial snake is inside the object, eventually it will expand and delineate the object boundary. DBC makes the force field anisotropic, which facilitates the initial snake contour to sense the actual object boundary [49]. We additionally use the directional gradient information here and we evolve the snake and compute the modified GVF field in an interleaved fashion until convergence. This intended modification makes the snake evolution more insensitive to initial snake location as well as it increases capturing capacity (it can reach to the object boundary starting from a small circle located inside an object) that we will demonstrate later. We call our modified GVF snake as Interleave directional Gradient Vector Flow (IDGVF) snake [49].
- We have proposed a novel pattern image for snake validation using PCA [46]. Our proposed pattern image is essentially constructed by the area of the original image covered by thickening the snake contour with an equal width both inside and outside. This pattern image is an annular ring across the snake contour. We warp this annular ring to a rectangular image for

computational convenience. This pattern image bears intensity transitions (dark-to-bright) across the object boundaries that characterize the object boundaries.

- We have also proposed a boosting based snake validation technique [49], where we have exploited the benefit of multiple features over any single feature (intensity transitions across actual object edges) used in the above PCA based snake classification. Experimental results demonstrate that boosting-based validation outperforms PCA-based validation technique. We have proposed a novel loss function for boosting that offers optimal weight on the misclassified samples at each stage of boosting that enforces misclassified samples to classify correctly in the following iterations and thus promotes boosting convergence. A user can regulate the amount of regularization through our proposed loss function. We derive a slightly modified Adaboost algorithm by minimizing the proposed loss function and we entitle our modified Adaboost algorithm as "Adaptively Regularized boosting" (ARboost). Experimental results demonstrate that ARboost can classify snake contours more accurately than other variants of boosting.

1.4 Organization of the thesis

The rest of the thesis is organized as follows. Chapter 2 presents an overview to the relevant background information regarding previous endeavor towards snake automation. Chapter 3 presents our proposed method towards snake automation. Chapter 4 discusses proposed snake initialization techniques. Proposed snake evolution technique is explained in Chapter 5. Chapter 6 provides proposed snake validation technique. Experimental results are illustrated and discussed in Chapter 7. Chapter 8 concludes this thesis. Then it follows with appendix and bibliography.

Chapter 2

2. Background and Related Work

Since their inception by *Kass et al*. in 1987 [21], active contour or snake has drawn much attention in numerous computer vision and image processing applications, particularly to locate object boundaries. Active contour or snake is essentially a 2-D parameterized curve defined within an image domain that evolves from an initial position toward the boundary of an object by minimizing the energy functional,

$$E_{snake} = E_{internal} + E_{external}, \tag{2.1}$$

Where,

$$E_{internal} = \frac{1}{2}\int_0^1 \{\alpha[(\frac{dX(s)}{ds})^2 + (\frac{dY(s)}{ds})^2] + \beta[(\frac{d^2X(s)}{ds^2})^2 + (\frac{d^2X(s)}{ds^2})^2]\}ds \tag{2.2}$$

and

$$E_{external} = -\int_0^1 f(X(s),Y(s))ds, \text{ where } f(x,y) = |\nabla I(x,y)|^2 . \tag{2.3}$$

Here, the contour is parameterized by $s \in [0, 1]$, and the coordinates of a point on the contour are ($X(s)$, $Y(s)$). I(x, y) denotes intensity values at pixel (x, y) and ∇I represents gradient of I. The internal energy defined in equation (2.2) is composed of a first-order term controlled by α and a second-order term controlled by β. The first order term forces the contour to be continuous by minimizing the distance between two consecutive points lying on the contour. However, it has the effect of causing the contour to shrink. The purpose of the second order term enforces smoothness and avoids oscillations of the snake by penalizing high contour curvatures. Setting the value of β to zero at a point allows the snake to become second-order discontinuous and develop a corner. The external energy attracts the contour toward the target boundary. The snake exploits *a priori* knowledge of object shape and inherent smoothness, usually formulated as internal energy to compensate for noise, gaps and other irregularities in object boundaries. The value of the external energy becomes very small when the snake points are closer to an edge. Snake achieves a minimal energy or equilibrium state when the curve reaches the targeted image boundaries and eventually locks onto the object boundary or edge without any further movement.

To minimize (2.1), calculus of variations is applied to obtain the following Euler equations [39]:

$$-\alpha\frac{d^2X}{ds^2}+\beta\frac{d^4X}{ds^4}-\frac{\partial f}{\partial x}=0, \qquad -\alpha\frac{d^2Y}{ds^2}+\beta\frac{d^4Y}{ds^4}-\frac{\partial f}{\partial y}=0. \tag{2.4}$$

To solve (2.4), (X(s), Y(s)) are treated as a function of time: (X(s, t), Y(s, t)), and the solutions to (2.4) are obtained when the steady state solutions of the following gradient descent equations are reached starting from an initial contour (X(s, 0), Y(s, 0)):

$$\frac{\partial X}{\partial t}=\alpha\frac{d^2X}{ds^2}-\beta\frac{d^4X}{ds^4}+\frac{\partial f}{\partial x}, \qquad \frac{\partial Y}{\partial t}=\alpha\frac{d^2Y}{ds^2}-\beta\frac{d^4Y}{ds^4}+\frac{\partial f}{\partial y}. \tag{2.5}$$

Equation (2.4) can be viewed as a force balance equation, so that if (u(x, y), v(x, y)) is any force field (computed from image data) acting on the active contour, then instead of solving equation (2.5) one solves the following equations,

$$\frac{\partial X}{\partial t} = \alpha \frac{d^2 X}{ds^2} - \beta \frac{d^4 X}{ds^4} + u, \qquad \frac{\partial Y}{\partial t} = \alpha \frac{d^2 Y}{ds^2} - \beta \frac{d^4 Y}{ds^4} + v. \tag{2.6}$$

The force field (u, v) is termed as the external force field. In (2.5), (u, v) is the edge force (f_x, f_y), *i.e.*, the gradient of the edge map, f, as defined in (2.3).

Traditional snake based image segmentation technique is performed in two steps: first, the user draws the initial contour of the snake and then the snake converges onto the desired object boundary guided by the force as defined in (2.6). There are two major drawbacks of this traditional snake model. First, because of limited capture range or poor convergence the snake cannot converge onto the desired object boundary, if the initial placement of the snake contour is away from the edges, because the force field is present only near the object boundaries or edges and this force diminishes away thereafter. Secondly, during initialization it is very difficult to find seed points automatically within the objects since clutters present in the noisy images mimic real objects. Thus, automatic initialization techniques lead to over segmentation, *i.e.*, number of seeds are far greater than the number of objects. To overcome these difficulties several methods have been proposed in literature. We discuss some noteworthy past endeavors for resolving these two difficulties below.

2.1 Past researches / endeavors on enhancing capture range of traditional snakes

One of the limitations of the traditional snake formulation [21] is its limited capture range. The external forces die out very rapidly away from the object boundary. If the initial active contour is away from edges and resides in a homogeneous region in an image, then the external force model (2.3) cannot attract the contour towards object boundary. Because, inside the homogeneous region, the image gradient magnitude would be zero and there will be no edge force (f_x, f_y) acting on the snake. Thus being guided only by the internal force, the active contour may not move towards desired edge. If there is no external force, the snake shrinks on itself and vanishes to a point or straightens to a line depending on the boundary conditions. We mention here a few significant researches conducted to increase the capture range of the tradition snake model: (a) Pressure force / Balloon snake [5], (b) GVF [56] and (c) Enhanced Generalized Gradient Vector Flow (EGGVF) [36].

(a) Pressure force / Balloon snake

Cohen [5] proposed another type of external force for active contour called pressure force / balloon snake that significantly increases the capture range of a traditional snake. The balloon force tries either to inflate or to deflate a closed contour. The balloon force exerts a force that is normal to the active contour (outward or inward). If it is an inflating force then the direction is outward normal; otherwise, it is directed to the inward normal. The balloon force for inflation is defined by,

$$F_{balloon} = k_1 \overrightarrow{n}(X(s),Y(s)) - k\frac{\nabla f_1(X(s),Y(s))}{\|\nabla f_1(X(s),Y(s))\|}, \textit{ where } f_1(x,y) = -|\nabla I(x,y)|^2 \text{ .} \quad (2.7)$$

Where $\overrightarrow{n}(X(s),Y(s))$ is the normal unit vector to the curve at point ($X(s)$, $Y(s)$), k_1 is the amplitude of this force and k is a user defined parameter. For deflation, the sign of k_1 will be negative. The balloon force enforces to keep force vectors of large magnitude away from the object boundary. Although the balloon force increases the magnitude of the forces it may fail to converge at boundary concavity [56]. One major bottleneck of the balloon force is that it is quite sensitive to the image edge strength. Snakes guided by pressure forces often leak through weaker edges [39].

(b) Gradient Vector Glow

Xu and Prince [56] proposed Gradient Vector Flow (GVF) that represented a noteworthy advance in snake formulation. In GVF, an external force field (*u*(*x*, *y*), *v*(*x*, *y*)) is constructed by diffusing the edge force (f_x, f_y) into the homogeneous regions, at the same time keeping the constructed field as close as possible to the edge force near the edges. This goal is achieved through the minimization of the energy functional,

$$E_{GVF}(u,v) = \frac{1}{2}\iint \mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) + (f_x^2 + f_y^2)((u - f_x)^2 + (v - f_y)^2)dxdy, \qquad (2.8)$$

where, μ is a non-negative parameter expressing the degree of smoothness of the field (*u*, *v*). u_x, u_y, v_x, v_y are the derivatives of *u* and *v* with respect to *x* and *y* respectively. *f* is the edge-map defined in (2.3). Minimization of (2.8) leads to the solution of classical Laplace's equation $(\nabla^2 u = 0 \text{ or } \nabla^2 v = 0)$ due to the first integrand in (2.8) that encourages making the field (*u*, *v*) smooth. Minimization of the second integrand forces the vector field to be close to the edge force near the edges where the edge force strength is high. μ is a regularization parameter that makes a balance between the first and second term in the integrand and the value of this parameter should be set considering the amount of the noise present in the image. A larger value of μ can suppress noise to a greater extent.

Minimizing (2.8) with the help of calculus of variations results in the following two Euler equations [56]:

$$\mu\nabla^2 u - (f_x^2 + f_y^2)(u - f_x) = 0, \qquad \mu\nabla^2 v - (f_x^2 + f_y^2)(v - f_y) = 0, \tag{2.9}$$

Solving (2.9) for external force field (u, v) results in GVF that acts as an external force field for snake.

The value of the second term in each equation of (2.9) is zero in a perfectly homogeneous region because $I(x, y)$ is constant in a perfectly homogeneous region and consequently its derivative is zero. However, in this scenario, u and v are each determined by Laplace equation (first term of each (2.9)), and the resulting GVF field is interpolated from the region's boundary. Thus, GVF exists in homogeneous regions (*i.e.*, where edges are absent) as well, and successfully increases the capture range of the edge force. Although GVF possesses many desirable properties as an external force field for snakes, it encounters difficulties to drag active contour inside a very narrow long concavity because it obliterates external force vector inside a very narrow long concavity while performing smoothing operation. Towards achieving this goal, a slight variation of GVF, called generalized gradient vector flow (GGVF), has been proposed [57]. The energy functional for GGVF is defined as,

$$\begin{aligned} E_{GGVF}(u, v) = \frac{1}{2}\iint & g(|\nabla f|)(u_x^2 + u_y^2 + v_x^2 + v_y^2) + \\ & (1 - g(|\nabla f|))((u - f_x)^2 + (v - f_y)^2)dxdy, \end{aligned} \tag{2.10}$$

where, g is a decreasing function of the edge force magnitude, which is defined

by, $g(|\nabla f|) = \exp(-(\frac{|\nabla f|}{k}))$, (2.11) with *k* as non-negative smoothing parameter for the field (*u*, *v*). Unlike the GVF functional (2.8), the GGVF functional (2.10) smoothes the force field (*u*, *v*) only when the edge force magnitude is very low. Minimizing (2.11) with the help of calculus of variations results in the following two Euler equations [57]:

$$g\nabla^2 u - (1-g)(u - f_x) = 0, \qquad g\nabla^2 v - (1-g)(v - f_y) = 0. \qquad (2.12)$$

(c) Vector Field Convolution (VFC)

The idea behind gradient vector flow is quite attractive – to diffuse vector field away from the object boundaries that enhances the capture range of the snake. Li *et al.* [23] proposed vector field convolution (VFC) where they utilized the idea of GVF by computing the diffused external force via convolution in real time. In VFC, edge map is convolved with a prefixed vector kernel that smoothes gradient vector field, makes vector field more robust to noise and it reduces computational cost. The VFC external force $f_{vfc}(x,y) = [u_{vfc}(x,y), v_{vfc}(x,y)]$ is given by calculating the convolution of the vector field kernel *k*(*x*, *y*) and the edge map *f*(*x*, *y*) generated from the image *I*(*x*, *y*), $f_{vfc}(x,y) = f(x,y) * k(x,y) = [f(x,y) * u_k(x,y),\ f(x,y) * v_k(x,y)]$. Here, $k(x,y) = [u_k(x,y), v_k(x,y)]$ is a vector field kernel in which all the vectors point to the kernel origin $k(x,y) = m(x,y)\,\mathbf{n}(x,y)$. *m*(*x*, *y*) is the magnitude of the vector at (*x*, *y*) and **n**(*x*, *y*) is the unit vector pointing to the kernel origin (0, 0), $\mathbf{n}(x,y) = [-x/r,\ y/r]$ except that $\mathbf{n}(x,y) = [0, 0]$ at the origin, where $r = \sqrt{x^2 + y^2}$ is the distance from the origin. The origin is considered as the Features of Interest (FOI) such as object edges so that a free particle placed in the field is able to move to the FOI. The VFC field highly depends on the magnitude of the vector field kernel *m*(*x*, *y*). It is assumed that the influence from the FOI should decrease as the particles are further away, the magnitude should be a decreasing positive function of distance from the origin. Li *et al*. [23] proposed two types of magnitude functions as follows:

$(a)\ m(x,y) = (r+\varepsilon)^{-\gamma}; (b)\ m(x,y) = \exp(-r^2/\tau^2)$, where γ and τ are positive parameters to control the decrease, ε is a small positive constant to prevent division by zero at the origin. Since the edge map is non-negative and larger near the image edges, edges contribute more to the VFC than homogeneous regions. Therefore, the VFC external force will attract free particles to the edges.

(d) Enhanced Generalized Gradient Vector Flow (EGGVF)

Ray *et al*. [38] demonstrated that GGVF is quite sensitive to initial contour position and it fails to capture object contour if the initial active contour location is not carefully chosen. To overcome this difficulty, Ray *et al*. proposed enhanced GGVF (EGGVF) [36], which utilizes the prior knowledge about the location of the initial contour relative to the object in solving the GGVF equations. The prior knowledge is that the initial active contour is completely located inside the object to be captured. This prior knowledge is encoded in the form of a Dirichlet Boundary Condition (BC), which is essentially a constraint on the partial differential equation, (PDE)s (2.9) or (2.12). Ray *et al*. imposes Dirichlet BC in the GGVF-PDEs as follows:

$$g\nabla^2 u - (1-g)(u - f_x) = 0,\ g\nabla^2 v - (1-g)(v - f_y) = 0,\ \text{when}\ (x, y) \in (D \setminus C)$$

$$(u, v) = n_{\partial C}\ \text{when (x, y)} \in \partial\text{C}, \nabla\text{u.n}_{\partial D} = 0, \quad \text{and} \quad \nabla\text{v.n}_{\partial D} = 0, \tag{2.13}$$

where, *D* denotes the rectangular image domain, *C* denotes the circular domain enclosed by the initial snake, *D*\C denotes the set difference of *D* and *C*, ∂D and ∂C are respectively the boundaries of *D* and *C*, $n_{\partial C}$ and $n_{\partial D}$ are unit outward normals to the boundaries ∂C and ∂D respectively. *g* is defined in (2.11) and *f* is defined in (2.3). The Dirichlet BC in (2.13) is characterized by imposing the solution as the unit outward normal on the boundary. The PDE (2.13) is utilized to generate EGGVF. Although the EGGVF has some similarity to the pressure force / balloon snake, the later is quite sensitive to the image edge strength. Pressure force / balloon snake has a tendency to leak through the weak edges whereas the snake guided by EGGVF force recovers the strong as well as the weak intensity edges [39]. The EGGVF-PDE smoothly interpolates the solution between the BC and the gradient of the edge-map. Thus the EGGVF-PDE can perceive the edge strength, whereas snake guided by balloon force cannot anticipate the gradient magnitude in its path.

Figure 2.1 shows the results of traditional snake proposed by Kass *et al*. [21], GVF [56] and EGVVF [36] on a synthetic circle image. A small circle as shown by dotted line is placed near to the circle boundary. Result shows that only EGGVF can capture circle contour successfully whereas traditional snake, GVF and VFC collapses on one side of the contour. The reason is visually evident from the force field of the corresponding snakes. For a traditional snake, the force field becomes feeble within the object and the resultant force acting on the small circle does not guide the snake to expand and capture the circle contour. Since GVF and VFC field are isotropic, in order to capture contour with a GVF and VFC snake, the initial contour should include circle center from where GVF and force arrows radiate as shown Figure 2.1. Otherwise, the active contour only faces the arrows all directed away from the center and it collapses at one side of the object boundary as illustrated in Figure 2.1. However, enhanced GGVF force field can sense the position of the initial contour inside the object and consequently it guides the contour to converge at object boundary. We have tailored EGGVF snake for oil sand particle delineation and leukocyte detection that has been explained in chapter 5.

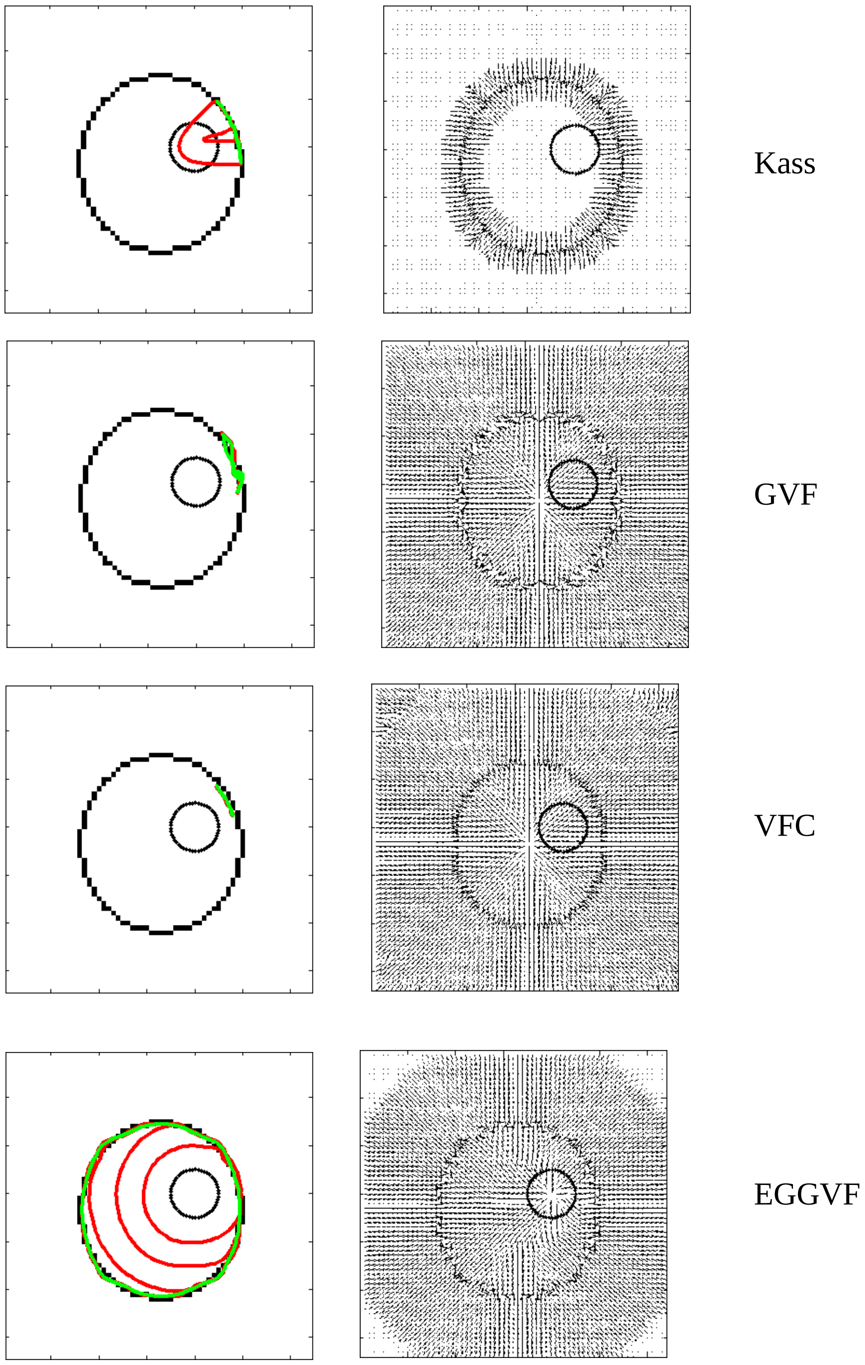


Figure 2.1: Left column: black circle shows initial contour; red circle shows intermediate contour and solid green circle shows fully evolved contour. Right column: external force field of the corresponding evolution algorithm.

2.2 Literature review on automatic snake initialization

The snake energy minimization is performed in two steps: first, the contour is placed near the object boundary and then the contour is evolved so that the energy decreases gradually and finally achieves the local minima of the energy functional. The energy functional is designed in such a way that the local minima are achieved when the snake contour locks onto the boundary. Although considerable amount of researches have been conducted on enhancing the capture range of the snake so that it can capture the accurate object boundary evolved from an initial seed located away from the target boundary, success of snake algorithm still depends on the initial position of the active contour.

Initialization plays a major role in the final solution quality of the snake model. One of the popular ways to initialize snake is employing some simple geometric shapes, such as circle or rectangle [22]. The potential bottlenecks of this simple initialization principle include a large number of iterations required for snake convergence and also converge away from object boundaries. An effective alternative approach is interactive segmentation [21], [56] where the user draws rough object outlines near the object boundaries and allows snakes to evolve from these initial contours and finally lock onto object boundaries. However, manually drawing object outlines over many images are extremely tedious, time consuming and overall not a cost effective process [22]. Since the introduction of GVF external force, various researchers have contributed on automatic snake initialization by analyzing the GVF external force field. Here, we discuss a few significant contributions.

(a) Center of Divergence (COD)

Ge and Tian [13] proposed an automatic initialization method for detecting multiple objects by placing small circles at centers of divergence (COD). COD is defined as the set of points from which GVF vectors of all the neighboring pixels radiate as shown in Figure 2.2(b),(c).

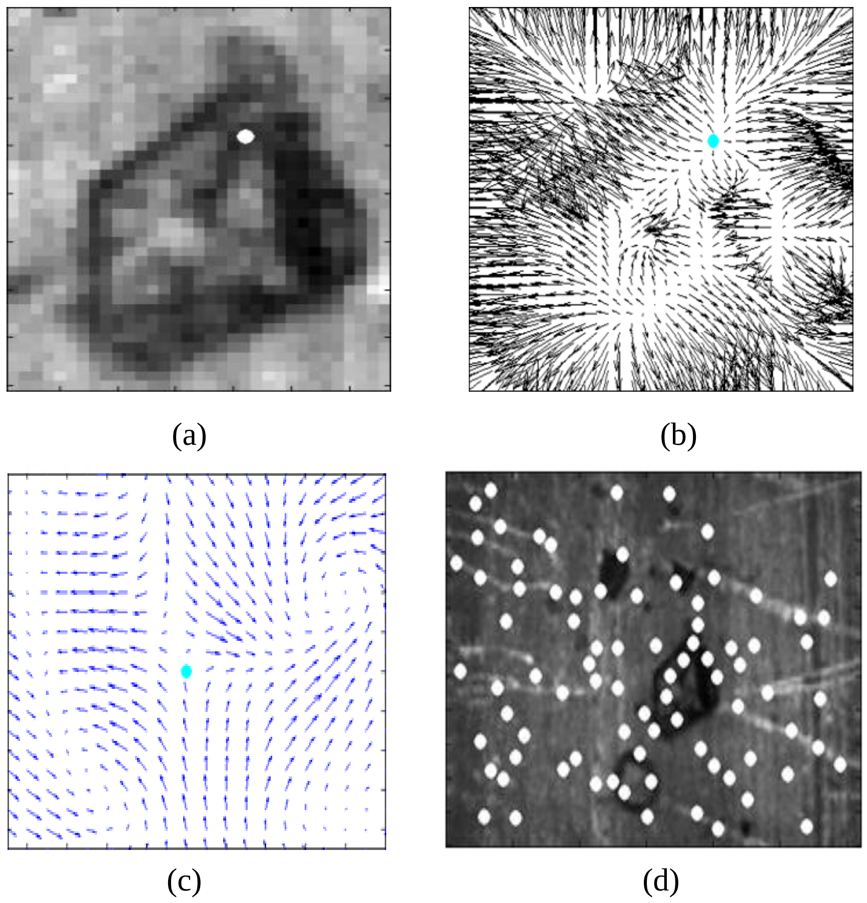


Figure 2.2: (a), (b) Marked point (dot) is a Center Of Divergence (COD) on an oil sand particle and GVF field; (c) Enhanced view of COD (dot) in GVF field displaying vectors of GVF field ;(d) Results of COD on Oil Sand image.

Let there be four pixels adjoining each other: $p\ (i, j)$, $p\ (i + 1, j)$, $p\ (i , j+1)$ and $p\ (i+1, j+1)$, where i and j denotes column and row number of pixel p. $V\ (i, j) = (x(i, j), y(i, j))$ is the corresponding GVF vector of the pixel $p\ (i, j)$. A function named *sign* is defined to quantify $x(i, j)$:

$$sign(x) = \begin{cases} 1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases},$$

$y(i, j)$ is quantified similarly. Then the potential scattering point set P_s is defined as follows:

$$P_{sx} = \{p(i, j) \mid x(i, j) \le x(i+1, j) \text{ and } abs(sign(x(i, j)) + sign(x(i+1, j))) \le 1\},$$

$$P_{sy} = \{p(i, j) \mid y(i, j) \le y(i, j+1) \text{ and } abs(sign(y(i, j)) + sign(y(i, j+1))) \le 1\},$$

$$P_s = [P_{sx}\ P_{sy}].$$

This potential scattering point set Ps is called the center of divergence (COD). COD refers to the local maxima of the external energy field.

(b) Critical point (CP)

Wang *et al.* [54] suggested the idea of critical point (CP), which is defined as the set of points from which the quantized GVF vectors of their 8-neighborhoods do not point to them as shown in Figure 2.3 (b), (c). Here the GVF vector is quantized in the following way: Let $\vec{v}_p$ be the associated GVF vector of a pixel p in the image domain, Ω_p be the 8-neighborhood of p, and let

q be another pixel in Ω_p. Further let $\vec{pq}$ be a unit vector from p to

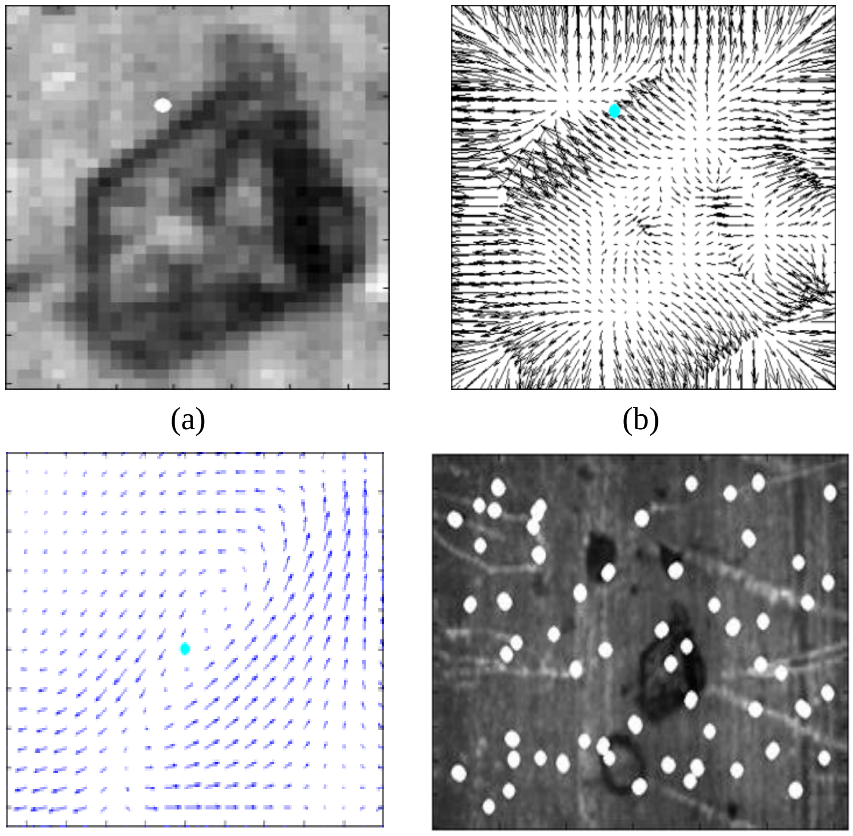


Figure 2.3: (a), (b) Marked point (dot) is a Critical Point (CP) on an oil sand particle and GVF field; (c) Enhanced view of CP (dot) in GVF field displaying vectors of GVF field; (d) Results of CP on Oil Sand image.

q. Then, $\vec{W}_p$ is derived from $\vec{v}_p$ such that, $\vec{V}_p.\vec{W}_p = \max_{q\in\Omega_p} \vec{V}_p.\vec{pq}$.Direction of $\vec{W}_p$ is the nearest to the direction of $\vec{v}_p$ among the eight $\vec{pq}$'s. Given a point p in the image domain, for any point q in Ω_p with $\vec{W}_q$, $\vec{qp}$ is a unit vector from q to p; p would be a critical point if, for all $q \in \Omega_p$, $\vec{W}_q.\vec{qp} < 1$. The point p is called *source* [58] or an inner critical if, for all $q \in \Omega_p$, $\vec{W}_q.\vec{qp} \le 0$. A source point resembles the origin of a spring where the water comes out and it is expected that the source point always lies within a homogeneous region *i.e.* inside an object.

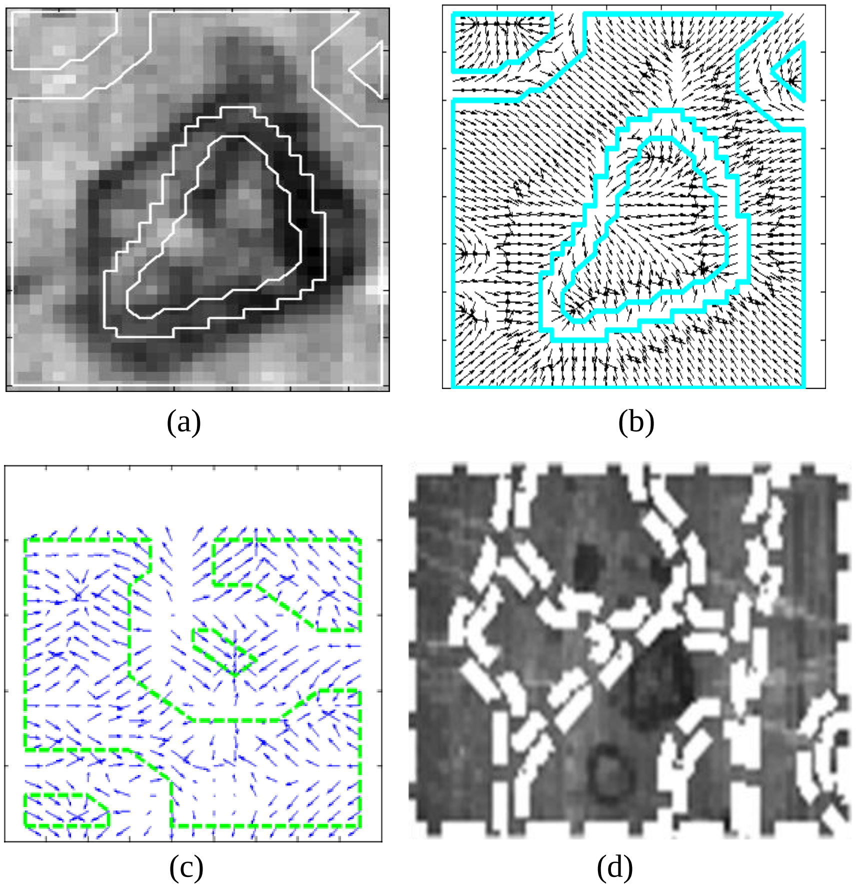


Figure 2.4: (a), (b) GVF field is segmented by Force Field Segmentation (FFS) shown by solid line on an oil sand particle and GVF field; (c) Enhanced view of FFS (solid line) in GVF field displaying vectors of GVF field; (d) Results of FFS on Oil Sand image.

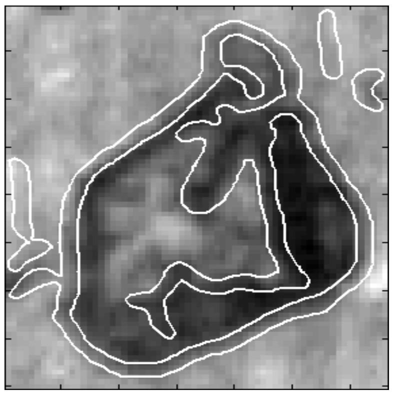
(c) Force field segmentation (FFS)

Li *et al.* [24] introduced an automatic snake initialization method via segmenting the external force field. FFS quantized the external force field similar to CP described above and treated edge map as a graph. Thus, FFS divides the external force field into different distinct regions by seeking weakly connected components in the graph in between opposing vectors as shown in Figure 2.4 (b), (c). Each region boundary is then initialized as an active contour. FFS is sensitive to clutter and broken edges and generally leads to spurious contours. FFS method cannot accommodate objects surrounded by extraneous edges. FFS is also a computationally expensive technique. However, FFS improves active contour performance by placing the simple shape initial model at the correct place; the initial model is still isolated from features of interest and requires a large number of iterations to converge [22].

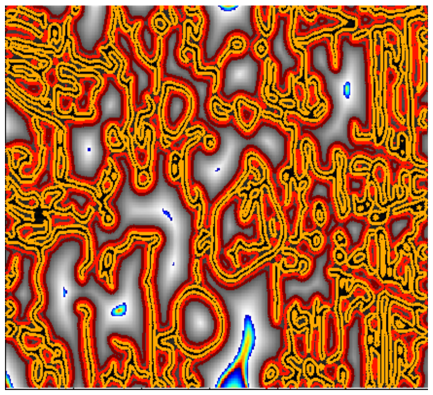
(d) Poisson Inverse Gradient (PIG)

Li and Acton [22] suggested Poisson inverse gradient (PIG) that estimates object boundary, which causes external force field. Traditional snake based segmentation algorithm first defines an external force field and subsequently evolves an active contour to delineate the object boundary. PIG instead solves the inverse problem where PIG determines the object boundary that produced a given external force field. PIG computes an external energy field associated with a given force field. Therefore, PIG seeks the energy $E(x, y)$ that has a negative gradient that is close to $f(x, y)$ in the L_2 norm sense: $E(x,y) = \underset{E}{\arg\min} \iint \left| -\nabla E(x,y) - f(x,y) \right|^2 dxdy.$ Minimizing this energy functional results in Poisson's equation: $\Delta E(x,y) = -div\ f(x,y)$. PIG finds the lines of constant value, termed *isolines*, within the energy field and defines the minimum isoline (isoline of minimum energy level) as snake initialization as shown in Figure 2.5. This constant value is known as isovalue. This minimal isoline represents a minimum energy approximation of the true optimal contour [40]. Isolines close to edges provide excellent candidates for snake initialization. Finding the optimal values of the parameters of these isomodels (isovalues of the isolines, number of isolines) is a difficult task that leads to over segmentation.

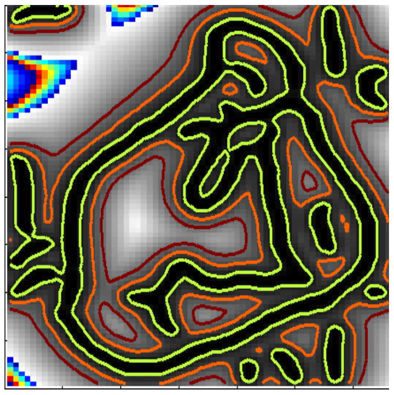
Results of the available automatic initialization techniques (COD, CP, FFS and PIG) on oil sand images (Figure 2.2 (d), 2.3(d), 2.4 (d) and 2.5 (d) respectively) illustrate that these methods lead to over-segmentation, *i.e.*, it initializes more than the desired number of snakes and region merging needs to be conducted on the image in the form of post processing. These demonstrate that these techniques are neither robust nor generic in nature and thus snake has been shown to be effective as a successful interactive image segmentation tool for the last two decades. From next chapter (chapter 3) onwards, we have discussed how we have tackled all these difficulties explained in this chapter (chapter 2) and have made effort on the development of a completely automated snake based image segmentation technique.



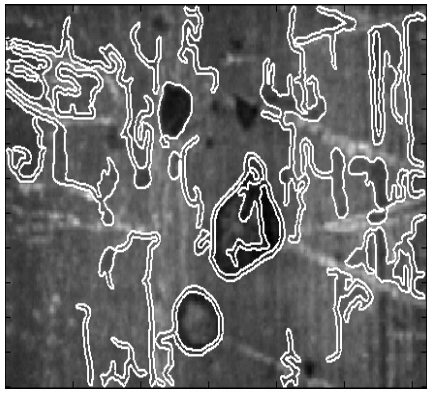
(a)



(b)



(c)



(d)

Figure 2.5: (a), (b) Isolines generated by Poisson Inverse Gradient (PIG) shown by solid line on an oil sand particle and GVF field; (c) Enhanced view of isolines (solid lines) on oil sand particle;(d) Results of PIG on Oil Sand image.

Chapter 3

3. Proposed Method for Snake Automation

We have illustrated in the previous chapter (chapter 2) that snake has been recognized as an interactive segmentation tool for the last two decades. However, snake fails to serve many significant image segmentation applications that require complete automation. Examples include oil sand particle delineation, leukocyte detection, and so on. This section depicts the proposed methodology towards snake automation.

Interactive snake segmentation algorithm consists of two sequential steps– (a) snake initialization: where a user draws rough objects outlines; and (b) snake evolution: where an energy functional is minimized to drive the user-drawn contour or snake to delineate the desired object boundary. Literature reveals that complete automation of snake consists of automating the snake initialization step. Unlike interactive snake segmentation process, seeds are placed here by an automated means and a snake is evolved from each seed. However, literature also shows that all automatic segmentation techniques suffer from over segmentation. We have first advocated in literature that for complete automation of snakes one needs to execute three sequential steps [47]: (a) snake initialization: where seeds are placed automatically on the image, (b) snake evolution: where one snake is evolved from each seed and (c) snake validation: where converged snakes are classified into object and non-object classes. Over-segmentation resulting from incorrect snake initialization could be compensated in the validation step as demonstrated in Figure 3.1.

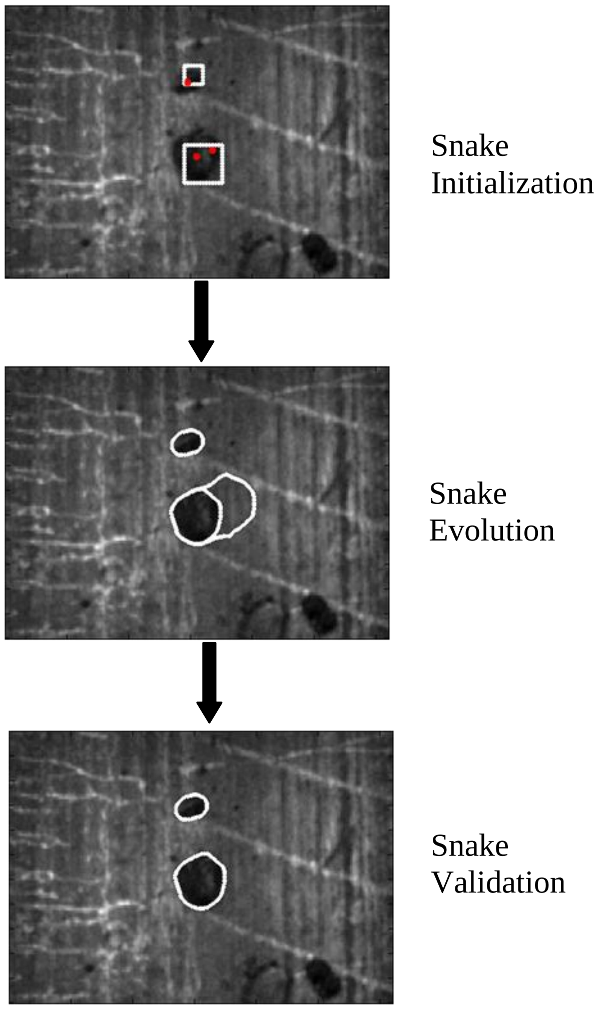


Figure 3.1: Proposed Snake based automatic segmentation technique.

Figure 3.1 illustrates three steps of proposed snake automation on an oil sand image where the snake initialization technique generates three seeds to segment two oil sand particles present in the image. Snake evolution technique evolves three snakes from three seeds. These three snakes are passed through a classifier where the classifier classifies two snakes as objects and the remaining one as non-object as shown in Figure 3.1. We have proposed two methods: Uniform Initialization (UI) and Probabilistic Quad Tree (PQT) based approximate snake initialization techniques for snake initialization, Interleave Directional Gradient Vector Flow (IDGVF) for snake evolution and two methods for snake validation: Principal Component Analysis (PCA) and Adaptive Regularized Boosting (ARboost) as shown in Figure 3.2.

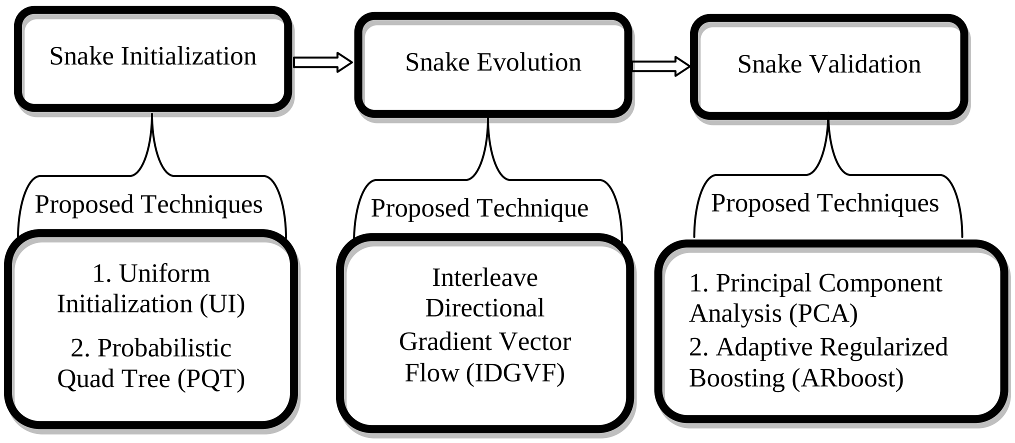

Snake Initialization
Snake Evolution
Snake Validation
Proposed Techniques
Proposed Technique
Proposed Techniques
1. Uniform Initialization (UI)
2. Probabilistic Quad Tree (PQT)
Interleave Directional Gradient Vector Flow (IDGVF)
1. Principal Component Analysis (PCA)
2. Adaptive Regularized Boosting (ARboost)

Figure 3.2: Three sequential steps of proposed snake based automatic segmentation technique and proposed methods for each of these three steps.

For Uniform Initialization (UI), we place seeds uniformly keeping uniform intervals between two consecutive seeds over the whole image as shown in Figure 3.3 (a). User can determine the separating distance between two consecutive seeds based on the object density. We place seeds uniformly at 30 pixels apart for oil sand particle delineation as shown in Figure 3.3 (a). Then we allow the snakes to evolve from each seed and finally classify each evolved contour into desired object and non-object classes using principal component analysis (PCA). To perform the contour classification, we construct a novel rectangular pattern image by warping an annular ring across each evolved snake contour. We project each pattern image into an already trained PC (principal component) space and compute PCA reconstruction error. The snakes associated with lower reconstruction errors (shown in Figure 3.3 (b)) than a threshold are identified as objects and others are identified as non-objects. Solid and dark lines in Figure 3.3(a) and Figure 3.3(b) represent oil sand particle and clutter respectively.

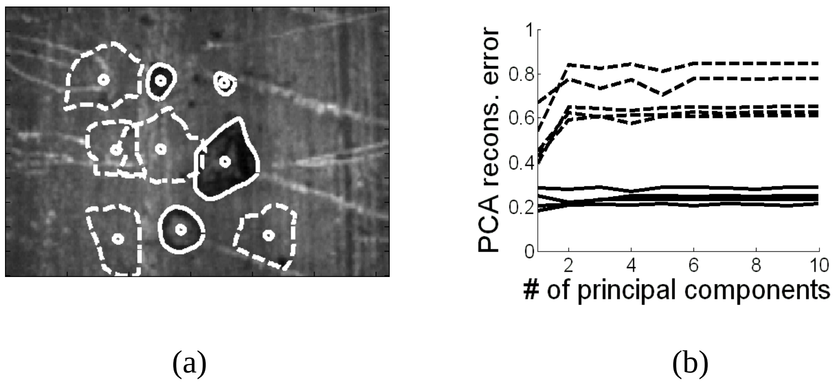


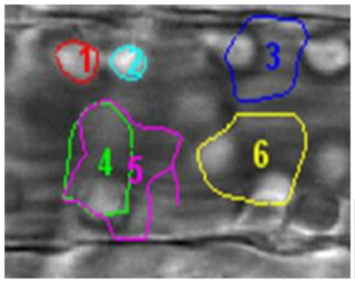
Figure 3.3: (a) Seed points (small circle) and evolved snakes. (b) PCA reconstruction errors of these snakes.

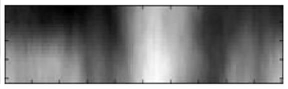
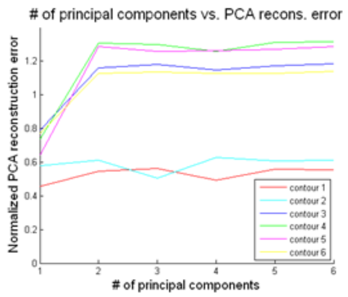
Placing seeds using UI over the whole image is not a computationally efficient method since snake evolution itself is a computationally expensive step. Over population of seeds translates into inefficiency of the overall segmentation algorithm. Thus, only few effective seeds are desirable. Towards achieving this goal, we propose a Probabilistic Quad Tree (PQT) based approximate segmentation technique, which finds regions of interest (ROIs) within an image as shown by white rectangles in Figure 3.1 and we place seeds using UI within the ROIs. If the ratio of the object area to the total object area is very high then UI alone is sufficient to use for snake initialization since PQT will not help much to reduce the number of seeds in this scenario. Details of proposed UI and PQT based snake initialization techniques are discussed in Chapter 4.

For snake evolution, we used Interleave Directional Gradient Vector Flow (IDGVF) snake. We have discussed in the previous chapter (chapter 2) that introducing Dirichlet Boundary Condition into Generalized Gradient Vector Flow (GGVF) makes the field anisotropic that facilitates the EGGVF [36] snake more resilient to initialization and also enhances the capture range. We have introduced two significant modifications into EGGVF framework. First, we have incorporated directional gradient information along with Dirichlet Boundary Condition that helps IDGVF to overcome the unwanted weak edges present inside the object and to lock onto actual object boundaries. We also compute the edge map in an interleaved fashion until convergence. This intended modification makes the snake evolution more insensitive to initial snake location as well as it increases capturing capacity (it can reach the object boundary starting from a small circle locating inside an object). More details of the proposed IDGVF snake evolution technique are explained in Chapter 5.

We have demonstrated PCA-based snake validation technique in Figure 3.1 and Figure 3.3. This PCA based snake validation technique is effective if prominent edges are present across actual object boundaries. We have also proposed a boosting-based snake validation technique that outperforms PCA-based validation. We have discussed these details in the Experimental Results and Discussions chapter (chapter 7). Boosting based validation exploits the advantages of using an effective combination of multiple features (edge, region and shape) over single edge based feature using PCA based snake validation technique. Details of the proposed PCA and boosting-based snake validation techniques are explained in Chapter 6.

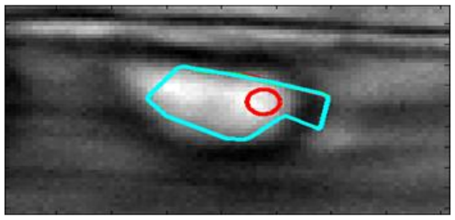
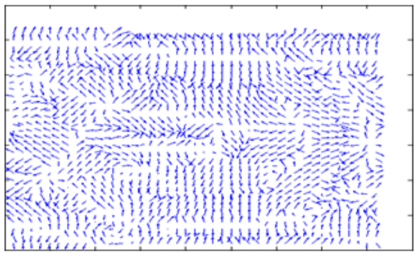
We have conducted experiments on two datasets: (a) oil sand mining images [59]: useful for improving the performance of oil sand mining; (b) leukocyte images [37]: helpful in the study of inflammation as well as in the design of anti/pro–inflammatory drugs. Results of our proposed algorithms have been demonstrated on oil sand images as shown in Figure 3.1 and Figure 3.3 as well as on leukocyte images as shown in Figure 3.4. Experimental results demonstrates in the Experimental results and discussions chapter (chapter 7) illustrate that our proposed algorithm is faster, more reliable and more accurate than the existing other competitive methods.



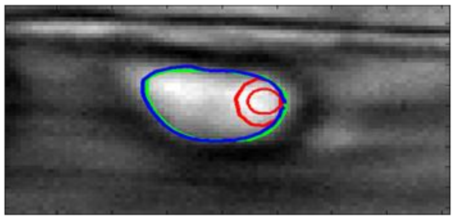
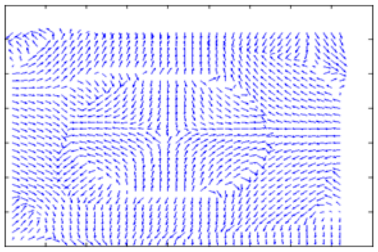



# of principal components vs. PCA recons. error
Normalized PCA reconstruction error
# of principal components
contour 1
contour 2
contour 3
contour 4
contour 5
contour 6

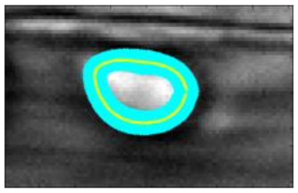
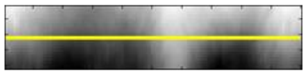
Uniform Initialization Pattern Image PCA reconstruction error

GVF Snake GVF Field

Proposed IDGVF IDGVF Field

Pattern image formation

Figure 3.4: Results of proposed snake based automatic segmentation technique on Leukocyte images.

Chapter 4

4. Snake Initialization

We have discussed in literature review (Chapter 2) that the existing automatic snake initialization techniques produce a lot of unsolicited seeds. Clutters available in the real images produce many imprecise seeds and snakes evolved from these seeds do not guarantee to converge snake contours onto the true object edges. Optimal number of snake evolution is a natural choice for multiple object detection using snake since snake evolution is an expensive step. We have proposed two snake initialization techniques: Uniform Initialization (UI) and Probabilistic Quad Tree (PQT) based approximate segmentation techniques. These two techniques exhibit less over segmentation than the existing snake initialization techniques without degrading the performance. Over-segmentation resulting from incorrect snake initialization could be compensated in the validation step. Proposed Uniform Initialization (UI) and Probabilistic Quad Tree (PQT) based snake initialization techniques are explained below.

4.1 Uniform Initialization (UI)

We place seeds uniformly over the whole image keeping at several pixel distances between two consecutive seeds and we name it as Uniform initialization (UI). We allow the snakes to evolve using IDGVF [46] from each seed. The principle of the proposed evolution (IDGVF snake evolution) is discussed in the next section. We have compared the performance of proposed UI with two existing snake initialization techniques, Center of divergence

(COD) [13] and critical point (CP) [54]. We recall that COD refers to the local maxima of the external energy field. COD is the set of points from which the GVF vectors of all the neighboring pixels radiate. CP is the set of points from which the quantized GVF vectors of its 8-neighborhood do not point to it.

Visual results of COD, CP and UI techniques on an oil sand image are shown in Figure 4.1. COD and CP yield 31 seed points on an average on each image, whereas we place 30 seeds on each image in UI. To compute performance evaluation metric (accuracy, recall, precision and F-measure) of snake initialization techniques (COD, CP and UI), we place seeds using each of the three initialization techniques on 100 oil sand images independently and then evolve one snake from each seed using IDGVF evolution. When snakes are fully converged, we validate converged snake contours into object and non-object classes using PCA that we have demonstrated chapter 6. We have kept snake evolution and snake validation techniques same for each of the three snake initialization techniques to make fair judgments on evaluating their performances. Accuracy, recall, precision and F-measure [41] for COD, CP and UI techniques are shown in Figure 4.2. F-measure is a harmonic mean of recall and precision and F-measure is defined as $2PR$ / (P+R), where P and R stands for precision and recall respectively. F-measure combines recall and precision into a single entity. A higher F-measure indicates better detection capability. Results show that UI with PCA validation technique recalls at least 45% better, detects at least 20% more precisely than C OD and CP. The

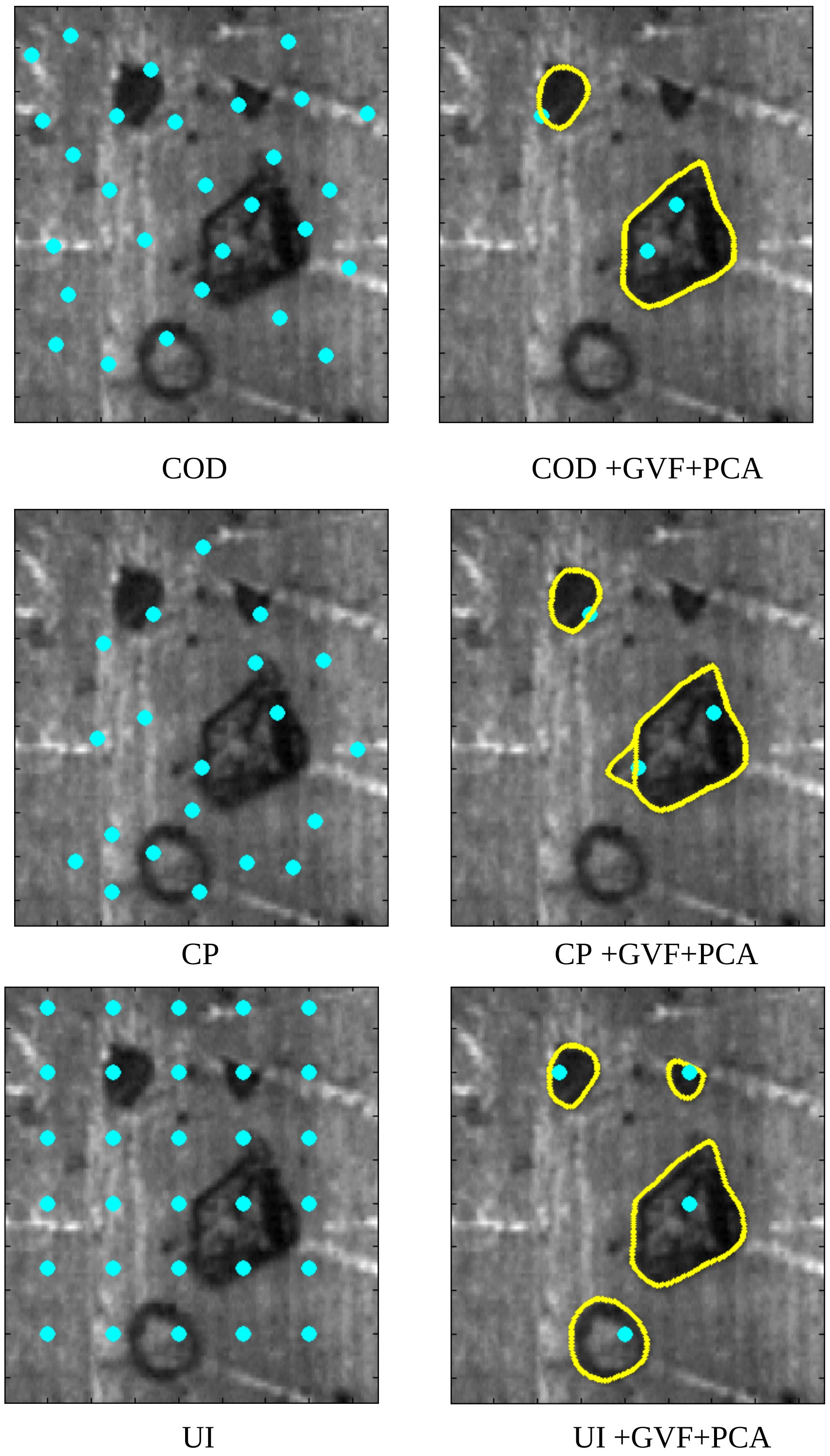


Figure 4.1: Snake validation by PCA for different snake initialization methods.

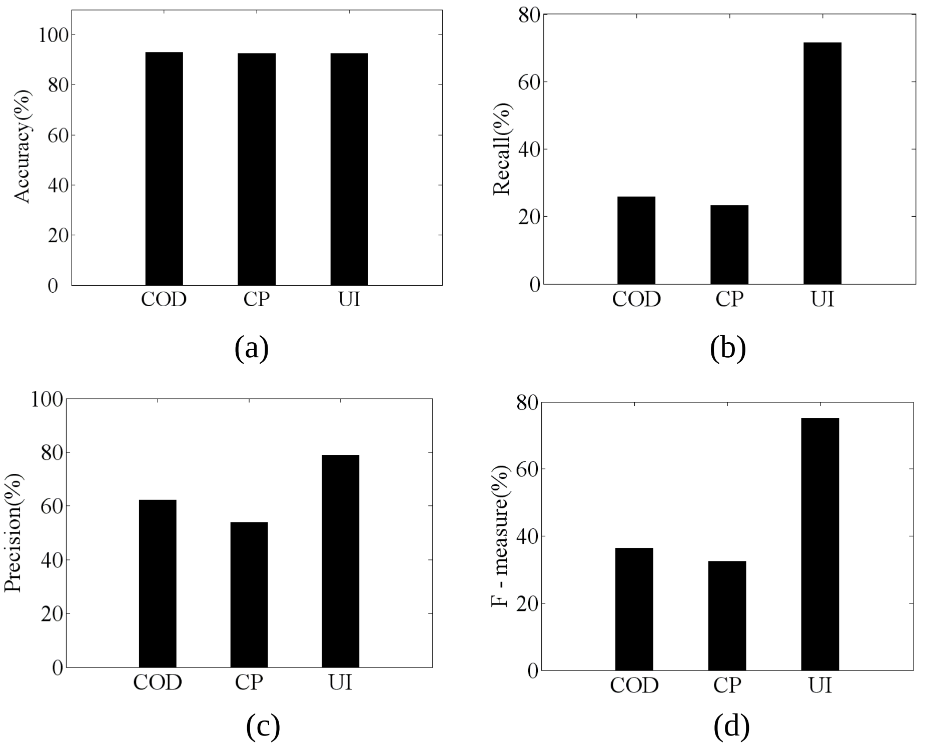


Figure 4.2: (a) Accuracy, (b) Recall, (c) Precision and (d) F – measure for Center of Divergence (CoD), Critical Point (CP) and Uniform initialization (UI).

value of F-measure for UI with proposed PCA technique is also bigger than those for COD and CP.

However, placing seeds over the whole image may not be appropriate for real-world applications, since the snake evolution is a computationally expensive step, and over population of seeds translates into inefficiency of the overall segmentation algorithm. To reduce the computational expense of the proposed UI technique, we have proposed probabilistic quad tree (PQT) based snake initialization technique. PQT first automatically finds regions of interest (ROI) from the image and then we place seeds using UI techniques within the ROIs. ROIs are the sub regions in an image where the probability of locating objects is higher than the rest of the image. We have also demonstrated in the Experimental results and discussions chapter (chapter 7) that PQT generates less number of seeds than the existing snake initialization techniques without impeding the performance and is therefore more efficient technique than other snake initialization techniques. The details of PQT are described in the next section.

4.2 Probabilistic Quad tree based approximate segmentation

PQT receives an image as an input, and then divides it into four adjacent, non-overlapping quadrants if it meets a pre-specified criterion, subsequently each quadrant is divided similarly and the process proceeds recursively until it fails the specified criterion. Consequently, the algorithm locates objects within rectangular boxes. The PQT algorithm is depicted in Table 4.1. PQT computes the ratio (r) of two posterior probabilities and splits the current region into four quadrants if r is between two predetermined thresholds: upper threshold (T_{up}) and lower threshold (T_{low}). We compute the ratio of these two posterior probabilities of a region (O) being object/non-object:

$$r = \frac{P(O=1/T,\, B)}{P(O=0/T,\, B)} = \frac{P(T/O=1)P(B/O=1)P(O=1)}{P(T/O=0)P(B/O=0)P(O=0)}, \qquad (4.1)$$

Where $P(T/O)$ and $P(B/O)$ are the likelihood of the region regarding texture and brightness respectively. O=1 signifies objects, O=0 encodes clutter. $P(O=1)$ is the prior probability of a region containing objects and $P(O=0)$ is the prior probability of a region including background. We locate objects by finding homogeneous regions based on local brightness and texture properties. We compute texture energy (T) by the distribution of the responses of Gabor filters

[30] and brightness (B) by the distribution of the gray level intensities (I) [33] of the region. Gray level intensity distribution determines brightness and Gabor filter response represents discriminative texture information for the objects. It is assumed that $P(T/O=1)$ and $P(B/O=1)$ follow multivariate normal distribution.

```
Input: Image ($I$)      *** $I$ is an $m$ by $n$ image matrix containing
***  gray level intensity;

Output: a set of disjoint patches, $R_i$, where $i = 1, 2, \ldots, n$ and
where $\bigcup_{i=1}^{n} R_i = I, R_i \bigcap_{i \neq j} R_j = \{\}$ (Null set) and Label ($R_i$) where
Label ($R_i$) is either 0 (for background) or 1(for foreground).

Initialzation: $k \rightarrow I$,     Label ($k$) $\rightarrow$ 0.
Call PQT ($k$) ***** PQT denotes Probabilistic Quad Tree ****
Algorithm PQT ($k$)

  1.  Compute $r$ using quation (4.1)
                          ***** $T_{low}$ and $T_{up}$ are two thresholds
  2.  If $r < T_{low}$          ***** $r < T_{low}$
  3.    $k \rightarrow$ background
  4.    Label ($k$) $\rightarrow$ 0
  5.  Else If $r > T_{up}$  ***** $r > T_{up}$
  6.        $k \rightarrow$ foreground
  7.        Label($k$) $\rightarrow$ 1.
  8.      Else            ***** $T_{low} < r < T_{up}$
  9.          Split $k$ into four disjoint quadrants, $k_1$, $k_2$, $k_3$, and $k_4$.
  10.         Quadtree($k_1$); Quadtree($k_2$);
  11.         Quadtree($k_3$); Quadtree($k_4$);
  12.     End
  13. End
```

Table 4.1: Probabilistic Quad-tree based approximate segmentation technique

However, it is unreasonable to make the same assumption for the background class since the object class contains only objects but the background class includes the rest of the world. Similar to the Bayesian discriminative features (BDF) [26] method, we derive a subset of the background classes that lie closest to the object class, and then model this particular subset of background class as a multivariate normal distribution. This statistical modeling of object and non-object likelihood is explained in section 4.3.

If $r > T_{\mathrm{up}}$, then the region is likely to contain objects and if $r < T_{\mathrm{low}}$, then the region is likely to be background. If $T_{\mathrm{low}} < r < T_{\mathrm{up}}$, the region is divided into four quadrants for further examination. The suitable values of T_{low} and T_{up} are found empirically as will be shown in the experimental section. Two threshold values (T_{low} and T_{up}) of splitting ratio (r) in PQT algorithm for oil sand images are determined as follows. First we estimate the prior probability and the parameters of the likelihood of object and background class from the training set containing 20 images. Then we draw bounding box along the objects to generate object patches and also choose background patches arbitrarily from the validation dataset and compute r for each of these patches. Normalized frequency *vs*. the value of r is plotted in Figure 4.3(a) for both object and background patches. We see that the value r for background patches lie between 0 to 20 and that for object patches lie from 0.6 to >30. We choose the values of T_{low} and T_{up} as 0.6 and 20 respectively for oil sand images as shown in Figure 4.3(a).

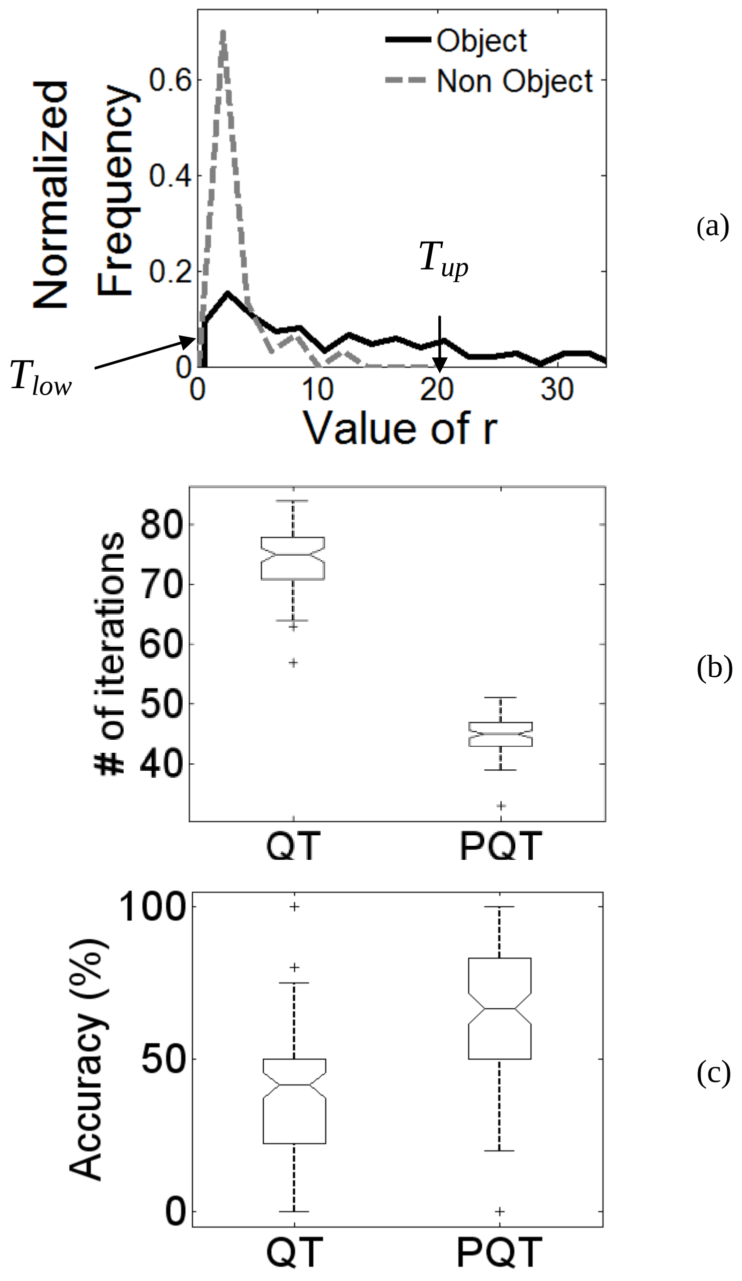


Figure 4.3: (a) Normalized frequency vs. value of the parameter *r* of PQT algorithm for object and non-object patches of oil sand images. (b) Number of iterations required for each oil sand image of QT and PQT algorithm. (c) Accuracy of QT and PQT algorithms.

We estimate prior probability of a region containing objects by computing the proportion of object pixels in the region from the training dataset. We estimate prior probability of a region including background similarly. This positional prior helps to converge PQT faster than Quadtree (QT) algorithm that we have demonstrated in Figure 4.3(b) and 4.3(c). Figure 4.3(b) illustrates that PQT converges after ~45 iterations whereas QT converges after ~75 iterations for an oil sand image. Since most of the oil sand particles are located on the central part of the image, positional prior prevents the non-central quadrants from splitting further and PQT terminates after fewer number of iterations than that of QT. At each iteration, QT divides a region into four quadrants if the mean average intensity and the mean response of the Gabor filter of the region are between the two thresholds values computed in a similar way to PQT. Two separate thresholds for brightness and texture are computed from the training set. Most importantly, Figure 4.3(c) demonstrates that PQT is more accurate than QT algorithm.

4.3 Statistical Modeling of Object and Non-object Classes

The object class contains only objects; the non-object class includes all the other objects, i.e., “the rest of the world” [26]. Even though it is reasonable to assume that the object class has a multivariate normal distribution, it is quite awkward to make the same assumption regarding non-object classes [26]. The response of the Gabor filters be T, is normalized by, $T = (\tilde{T} - \mu)/\sigma,$ where $T \in \Re^N$, μ and σ are the mean and the standard deviation of the components of T respectively. The conditional density function of the object class is modeled as a multivariate normal distribution:

$$P(T/O=1)=\frac{1}{(2\pi)^{N/2}|\Sigma_o|^{1/2}}\exp\{-\frac{1}{2}(T-M_o)^t\Sigma_o{}^{-1}(T-M_o)\} \quad (4.2),$$

Where $M_0 \in \Re^N$ and $\Sigma_0 \in \Re^{N\times N}$ are the mean and the covariance matrix of object class *O* respectively. The covariance matrix, $\sum_o$ can be factorized into the following form using the principal component analysis: with

$$\phi_0\phi_0^t = \phi_0^t\phi_0 = I_N, \Lambda_0 = diag\{\lambda_1, \lambda_2, ..., \lambda_N\} \quad (4.3),$$

Where $\phi_0 \in \Re^{N\times N}$ is an orthogonal eigenvector matrix, $\Lambda_0 \in \Re^{N\times N}$ is a diagonal eigenvalue matrix with diagonal elements (the eigenvalues) in decreasing order $(\lambda_1 \geq \lambda_2 \geq .... \geq \lambda_N)$, and $I_N \in \Re^{N\times N}$ is an identity matrix. The principal components are defined by the following vector, $Z \in \Re^N$:

$$Z = \phi_0^t\ (T - M_0) \quad (4.4)$$

From equation (4.2), (4.3) and (4.4) we have,

$$P(T/O=1)=\frac{1}{(2\pi)^{N/2}|\Sigma_o|^{1/2}}\exp\{-\frac{1}{2}Z^t\Lambda_o{}^{-1}Z\} \quad (4.5)$$

The components of *Z* are the principal components. We use only the first *M* $(M << N)$ principal components to estimate the conditional density function. We estimate the remaining *N* - *M* eigenvalues, $\lambda_{M+1}, \lambda_{M+2}, ..., \lambda_N$, by the average of those values:

$$\rho = \frac{1}{N-M}\sum_{k=M+1}^{N}\lambda_k \quad (4.6)$$

From (4.4) we have,

$$\|Z\|^2 = \|T - M_0\|^2, \tag{4.7}$$

Where $\|.\|$ denotes the norm operator. From (4.5) and (4.6) we have,

$$P(T/O=1) = \frac{1}{(2\pi)^{N/2}} \frac{\exp\{-\frac{1}{2}\sum_{i=1}^{M}\frac{z_i^2}{\lambda_i}\}}{\prod_{i=1}^{M}\lambda_i^{1/2}} \frac{\exp\{-\frac{\|T-M_0\|^2 - \sum_{i=1}^{M} z_i^2}{2\rho}\}}{\rho^{(N-M)/2}} \tag{4.8}$$

Where z_is are the components of Z defined by (4.4). Equation (4.8) states that the conditional densities function of object class can be estimated using the first M principal components, the input image, the mean of the object class, and the eigenvalues of the object class. The brightness likelihood of the object class $P(B/O=1)$ is derived in a similar way the texture likelihood computed above. $P(T/O=0)$ and $P(B/O=0)$ are derived similarly for non-object class also. For further details, please see [26], [31].

In the experimental section, we have demonstrated a comparative study between proposed PQT based snake initialization technique and other existing snake initialization techniques.

Chapter 5

5. Snake Evolution

In the literature review chapter (chapter 2), we have shown that Enhanced Generalized Gradient Vector Flow (EGGVF) is less sensitive to snake initialization than GVF since the EGGVF field can sense the position of the initial contour located inside the object by virtue of Dirichlet boundary conditions and it can guide the contour towards target boundary. However, EGGVF snake can be misguided by the weak edges as shown in Figure 5.1. Figure 5.1 demonstrates the evolution of an EGGVF snake from a seed located inside an oil sand particle. The corresponding edge map, vector field and intermediate stage of snake evolution are also illustrated in Figure 5.1. Edge map in Figure 5.1 shows the presence of weak and broken edges in the homogeneous region located inside the oil sand particle that is often found inside the object in reality.

Although EGGVF can evolve from a small initial contour and reach the actual object boundary, EGGVF snake can be misguided by weak edges (bottom left part of Figure 5.1) and unwanted strong edges (right top corner of Figure 5.1). To reduce the effect of weak and unwanted strong edges, we have modified EGGVF evolution as follows. First, we have incorporated directional gradient information along with Dirichlet boundary condition into GGVF energy functional that helps to capture object boundaries from a naïve initial contour (circle of radius around 2~3 pixels) located inside the object. Towards achieving this goal, our proposed snake algorithm first builds an edge map [46]:

$$\rho(x,y)=\max(0,\frac{(x-x_0)(\partial I/\partial x)+(y-y_0)(\partial I/\partial y)}{\sqrt{\left(x-x_0\right)^2+\left(y-y_0\right)^2}}),$$

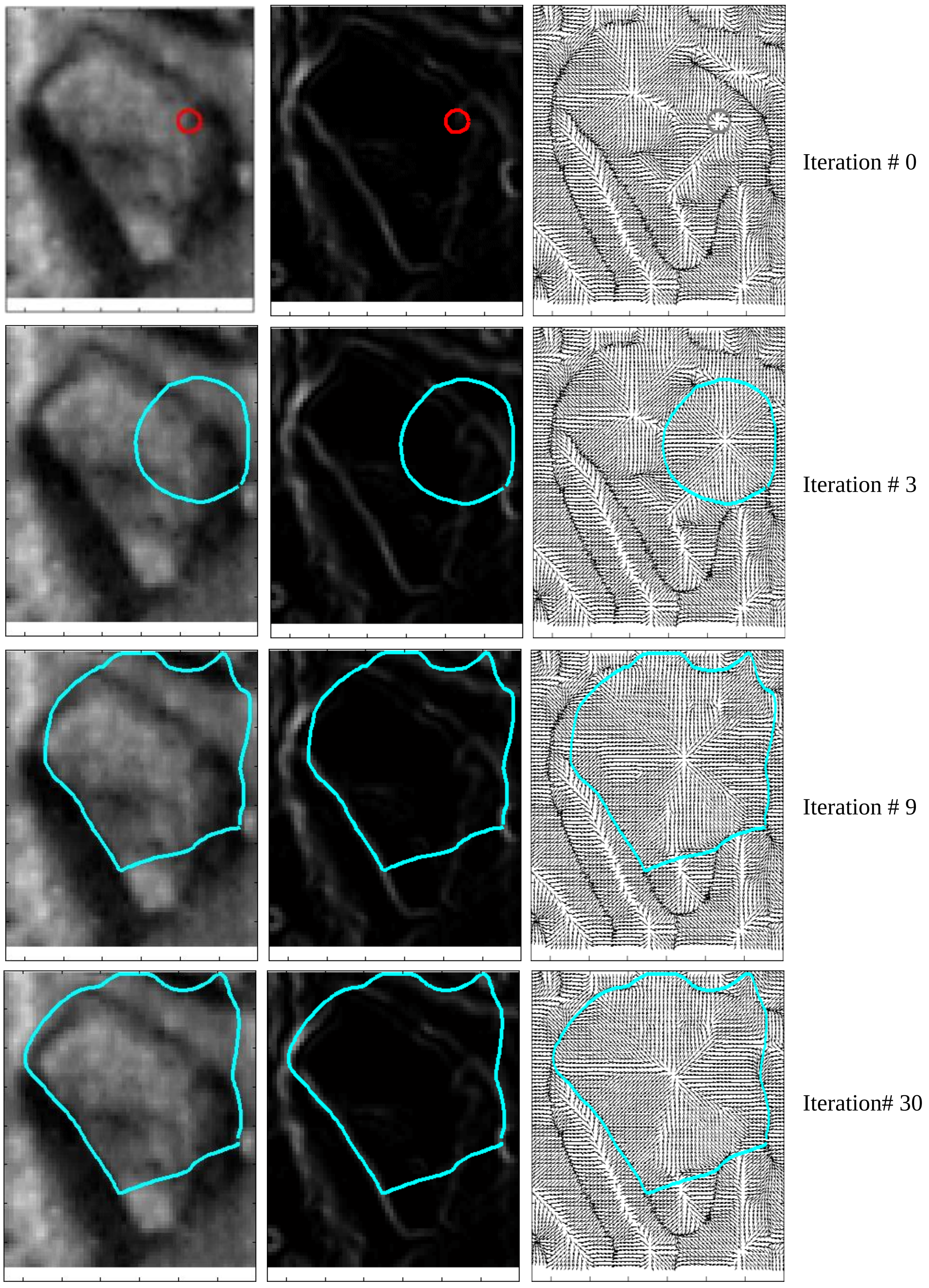
Where $I(x, y)$ is the image. The edge map ρ indicates only dark-to-bright intensity transitions on the image as seen from the center of the snake contour converged at previous iteration (x_0, y_0). Next, we compute a force field $(u(x, y), v(x, y))$ as follows [46]:

$$\partial u/\partial t=\exp(-\rho(x,y)/K)\nabla^2 u-(1-\exp(-\rho(x,y)/K))(u-\partial\rho/\partial x),$$
$$\partial v/\partial t=\exp(-\rho(x,y)/K)\nabla^2 v-(1-\exp(-\rho(x,y)/K))(v-\partial\rho/\partial y),$$

subject to the Dirichlet boundary condition:

$$(u(x,y),v(x,y))=\mathbf{n}(x,y),\ \text{for}\ (x,y)\in\partial\Omega,$$

Where $\partial\Omega$ is the initial snake contour (in our case, it is a small circle centered at (x_0, y_0)) and $\mathbf{n}(x, y)$ is the unit outward normal to the initial snake at (x, y). K is a user defined parameter controlling the degree of smoothness of the snake external force field (u, v). After computing (u, v), we use them to evolve a snake from the initial contour. Here the novelty is that we perform snake evolution and (u, v) computation in an interleaved fashion– first compute (u, v), then evolve a snake with (u, v) until convergence, next compute (u, v) again with previously evolved snake contour as $\partial\Omega$, and so on, until finally there is no appreciable change in the area enclosed by the snake. For the first iteration, (x_0, y_0) is the initial seed point around which a small circular initial contour is chosen for evolving the snake and from the second iteration onwards, (x_0, y_0) is



Evolution Edge Map Vector field

Figure 5.1: EGGVF snake evolution from a small naïve snake contour (red circle at Iteration # 0) chosen arbitrarily inside an oil sand particle.

the center of mass of the snake evolved in the preceding iteration. We call this variant of GVF as Interleave Directional Gradient Vector Flow (IDGVF).

For all other variants of GVF snake evolution available in literature, edge map is first built before commencing the snake evolution and this edge map is kept same until the snake is fully converged. However, in IDGVF, the directional edge map is newly constructed at each iteration since (x_0, y_0) is different for each iteration. IDGVF is a modified version of EGGVF [38]. For EGGVF, external force field is the image gradient vector. For IDGVF, the force field is the dot product of gradient vector and the unit vector formed by joining each point of the field with (x_0, y_0). The position of (x_0, y_0) changes at each iteration. Consequently, IDGVF can sense the actual object edges more accurately and it can surpass weak edges present inside the object due to image noise and eventually snake locks onto actual object edges evolving from a naïve initial contour inside an object. Figure 5.2 shows the evolution of IDGVF snake from a small initial contour placed arbitrarily inside an oil sand particle. The corresponding edge map and external force vector fields are also illustrated in Figure 5.2. One can conclude from the visual result that introduction of Dirichlet boundary condition into GGVF evolution framework enables snake to reach object boundary from a small initial contour by making the force field anisotropic. Introduction of directional gradient information and novel reconstruction of edge map at each iteration during snake evolution in an interleave fashion guide the snake to overcome weak edges located inside the object and sense actual object boundaries.

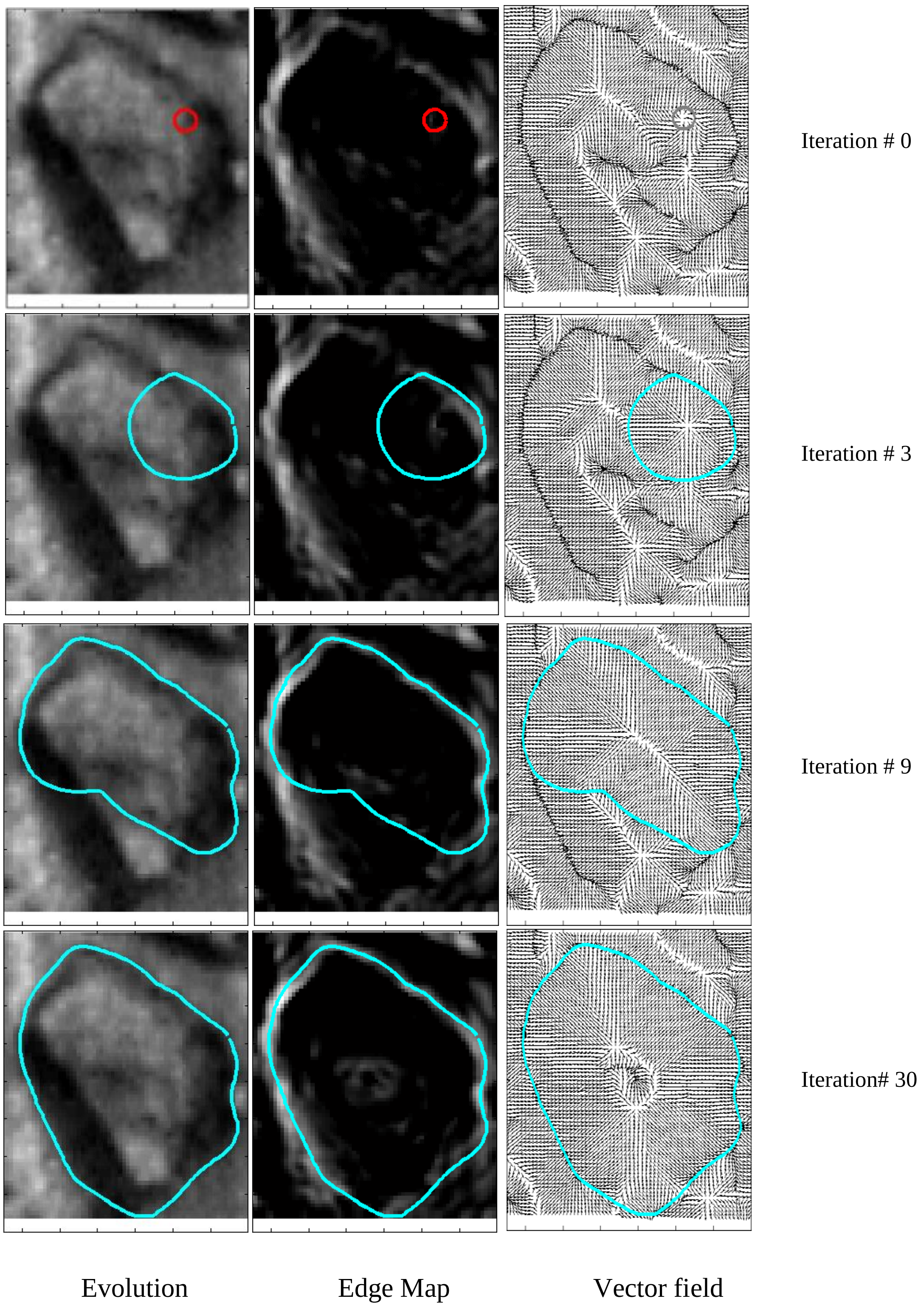


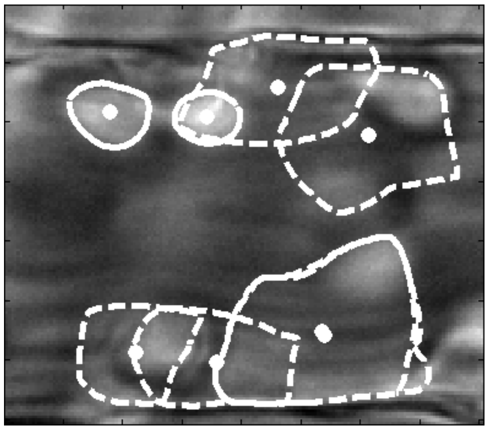
Figure 5.2: IDGVF snake evolution from a small naïve snake contour (red circle at Iteration # 0) chosen arbitrarily inside an oil sand particle.

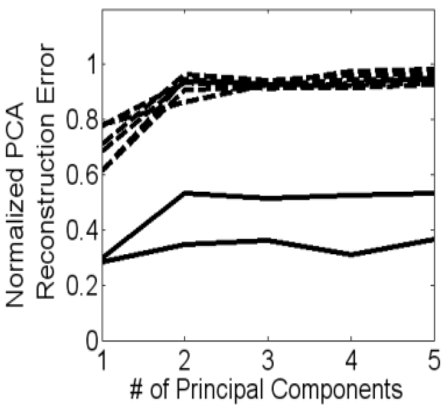
In the experimental results and discussions chapter (chapter 7), we have demonstrated that our proposed IDGVF snake evolution algorithm is quite robust to snake initialization compared to other snake evolution algorithms. It has a broad capture range and it can capture an object contour from a seed point located inside an oil sand particle. We have also presented in the experimental and discussions chapter (chapter 7) that proposed snake evolution algorithm can delineate blob objects more accurately than its competitors.

Chapter 6

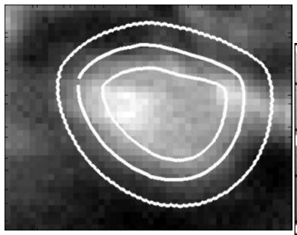
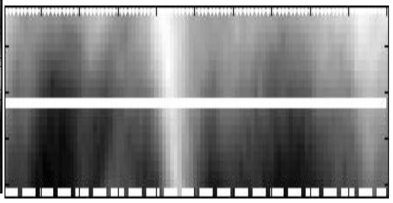
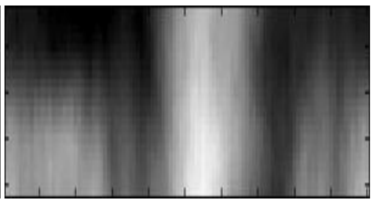
6. Snake validation

Literature reveals that snake automation technique consists of two sequential steps: snake initialization and snake evolution. Substantial efforts have been attempted to enhance the capture range of snake evolution. However, one of the main bottlenecks of snake automation is snake initialization since all of the snake initialization techniques suffer from over segmentation. We have first proposed in literature [47] that one needs to execute three sequential steps for snake automation: snake initialization, snake evolution and snake validation. We have demonstrated that snake validation is an important step for snake automation that has been unnoticed till date in literature [47]. Over-segmentation resulting from incorrect snake initialization could be compensated in the validation step. After complete convergence of snake contour from the given seed, the contour is classified into desired object and non-object classes in the validation step. We have proposed two novel snake validation techniques: Principal Component Analysis (PCA) based classifier and Adaptive Regularized boosting (ARboost) for snake validation. The details of these two validation methods are discussed below.


(a)



(b)

(c) (d) (e)

Figure 6.1: leukocyte detection by PCA-based snake validation proposed by Saha *et al*. [47]. (a) Four snakes evolved on a leukocyte image. (b) PCA reconstruction error computed on snakes of fig. (a). (c) Annual band (unwarped pattern image) around a snake (solid white line) is shown by dotted white line. (d) Warped (unfolded) pattern image for snake delineating a leukocyte is shown. (e) Warped (unfolded) pattern image for clutter.

6.1 Principal Component Analysis (PCA) based snake validation

We first place seed uniformly over the whole image as shown in Figure 6.1(a) where seeds are placed uniformly over a leukocyte image. White dots shown in Figure 6.1(1) are the position of the seeds. Then an IDGVF snake is evolved from each seed as shown in Figure 6.1(a). When all the snakes are fully converged, a novel pattern image is formed for each converged snake contour to classify snakes into object and non-object classes using principal component analysis (PCA). Pattern image is constructed by unfolding an annular band across the converged snake contour as shown in Figure 6.1(c), (d) and (e). The pattern image is projected into PCA space and PCA reconstruction error is computed. The pattern images associated with object classes demonstrate lower normalized PCA reconstruction error than pattern images associated with non-object classes as shown in Figure 6.1(b). Solid and dotted line in Figure 6.1(a) and (b) represent for leukocyte and clutter or non-object classes respectively. This PCA based validation technique is helpful if prominent edges (intensity transitions across the object boundaries) are present. The formation of pattern image and snake validation (classification of snake contours into object/non-object classes) by PCA using pattern image is discussed below.

6.1.1 Pattern Image Formation

Figure 6.2(a) shows a snake (in solid line) and an annular ring denoted by dotted contours inside and outside of the evolved snake contour for an oil sand particle. The pattern image is constructed by warping the annular ring into a rectangular image (Figure 6.2(b)). 100 sample points are considered uniformly across the contour to make same size of the pattern images irrespective of the object sizes. The pattern image associated with object class carries object texture information near the object boundary / intensity transition (dark to bright transition) across the object contour that has good discrimination capability. This intensity transition (dark to bright) is absent for non-object class as shown in (Figure 6.1(e)). This pattern image outperforms over two other gradient based techniques: Gradient Inverse Coefficient of Variation (GICOV) and average directional gradient strength that has been demonstrated in the experimental results and discussions chapter (chapter 7).

The width of the pattern image is chosen based on the object size and image intensity profile near the object boundary.

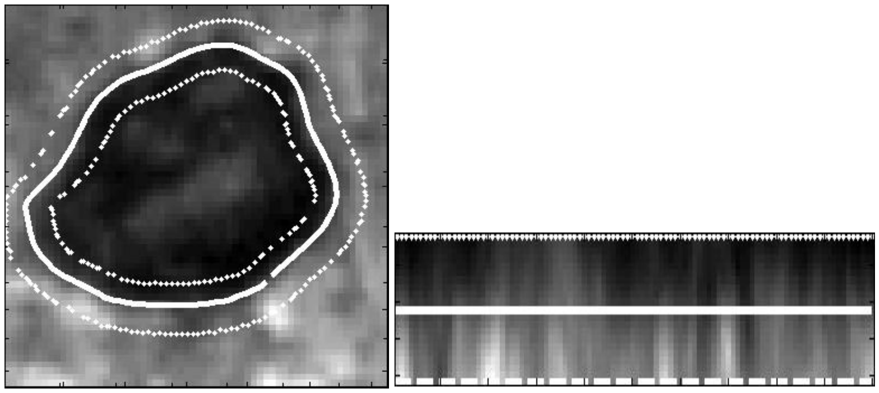

(a) (b)

Figure 6.2: (a) annular ring (from innermost to outermost curve) across the contour (middle curve) of the object. (b) Rectangular pattern image.

6.1.2 Snake Validation by PCA

Snake validation represents classifying converged snake contour into object and non-object classes. Given a set of training pattern images, the PCA framework for snake validation is as follows. Each pattern image I_i of size m-by-n is reshaped into a vector $\mathbf{x}_i$, of size M-by-1, with $M = mn$. Next, p pattern images are combined into a matrix: $[\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_p]$ of size M-by-p. Then, this matrix is centered: $X = [\mathbf{x}_1 - \bar{\mathbf{x}}, \mathbf{x}_2 - \bar{\mathbf{x}}, \dots, \mathbf{x}_p - \bar{\mathbf{x}}]$, where $\bar{\mathbf{x}} = (1/p)\sum \mathbf{x}_i$. Here typically, $M >> p$. Using singular value decomposition we have: $X_{M\times p} = U_{M\times p}\, D_{p\times p} (V_{p\times p})^T$, where, U's columns are eigenvectors of XX^T, and V's columns are eigenvectors of X^TX. D is a diagonal matrix with non-negative diagonal elements d_i, the singular values, which are the square roots of the eigenvalues λ_i of XX^T (or X^TX) and are usually ordered so that $\lambda_i \geq \lambda_{i+1}$, $i = 1, 2, .., p$-1. We compute the value of U, V and D from p pattern images associated with object class constructed from the training set. Then, we can project a new or test pattern image $\mathbf{x}$ into the PC-subspace composed of only $d << M$ eigenvectors (first d columns of $U_{M\times p}$): $\tilde{\mathbf{x}} = \bar{\mathbf{x}} + U_{M\times d}(U_{M\times d})^T(\mathbf{x} - \bar{\mathbf{x}})$. The value of d is computed from the magnitude of the corresponding eigenvalue λ. PCA reconstruction error for the test image $\mathbf{x}$ is defined as: $\|\mathbf{x} - \tilde{\mathbf{x}}\|^2$, where $\|.\|$ denotes the Euclidean norm. If the test pattern image $\mathbf{x}$ belongs to the same class as the training images, then this reconstruction error will be small; otherwise it will be large [18].

6.2 Snake Validation by Boosting

When snakes fully converge from the seed points on an image, we compute different features for each converged snake contour, such as, contour shape features [44] e.g., form factor, convexity, extent, modification ratio, major axis length, minor axis length, equivalent diameter, solidity, roundness, elongation, curl, aspect ratio, eccentricity, orientation, compactness, convex area [44], region based features e.g., area, intra and inter class variance, region instability [43] and edge based features e.g., GICOV [10], average gradient strength, smoothness, monotonicity [60], difference of Gaussian, residual error of the spin image [19] computed across the contour, temperature of the contour [7] , variance of the edge direction along the contour, edge life time [43], PCA reconstruction error [47] for snake validation. The definition of these features is given in the Appendix. We use proposed Adaptive Regularized Boosting (ARboost, a variant of boosting) for selecting important features. ARboost algorithm consists of two phases: training and testing.

6.2.1 ARboost training

At the training phase, ARboost selects only important features for snake validation from a large pool of features computed using training snake contours and finds the weights associated with those features. We place seeds uniformly over the training images, evolve one snake from each seed and classify the snakes manually as objects that converge at actual object contours; otherwise we consider the snakes as non-objects. Thus we are able to obtain a training set of positive and negative samples. The ARboost algorithm forms a strong classifier by combining a set of weak learners linearly in an iterative manner [15]. We use decision stumps [15] as weak classifiers. Decision stump is a single level decision tree. Decision stump, $G_j(x)$ for feature f_j is defined as, $G_j(x) = 1$ if $f_j(x) > \theta_j$; otherwise, $G_j(x) = 0$, where θ_j is some feature value of f_j chosen as a threshold. Finding the best decision stump at each stage is similar to learning a node in a decision tree. We search over all possible features $f = [f_1, f_2, f_3, \dots f_j, \dots, f_n]$ and for each feature, we search over all possible thresholds θ induced by sorting the observed values of f and pick f with θ that gives lowest misclassification error during training.

6.2.2 ARboost testing

At the test phase, the proposed PQT algorithm used for snake initialization finds ROIs automatically over the test images; then we place seeds uniformly only within ROIs and grow one snake from each seed. When all snakes fully converge, for each snake, we compute the values of the important features selected by ARboost during training phase and multiply them with the values of the weights associated with the features chosen by ARboost during the training phase and subsequently add them to form a strong classifier, $f(x) = sign(\sum_{m=1}^{M} \alpha_m G_m(x))$, where, α_m is the weight associated with weak classifier $G_m(x)$. If the sign of the strong classifier for a snake contour is positive then it is classified into object class, otherwise it is classified into non-object class.

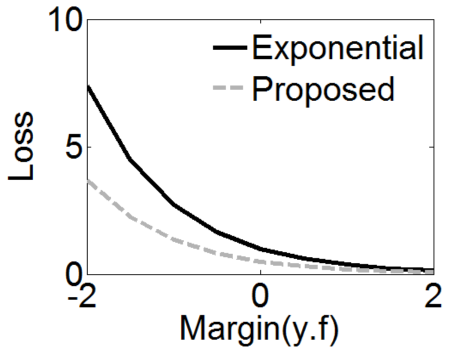


Fig. 6.3: Loss functions for two class classification.

For classification, standard Adaboost minimizes an exponential loss function: $L(y, f(x)) = \exp(-yf(x))$, where y is the response and f is the prediction. We introduce an additional term into exponential loss function: $L(y, f(x)) = \exp(-yf(x) + \lambda|y - G(x)|)$, where $\lambda < 0$ and $G(x)$ is the prediction of the weak classifier chosen at the current stage. In any boosting iteration, the proposed loss function is the same as the existing loss function if the misclassification error rate at current stage is zero (proposed term vanishes when $y = G(x)$). Proposed loss function estimates one-half the log-odds of P (Y = 1|x) similar to exponential loss criteria as well as binomial negative log-likelihood or deviance (also known as cross-entropy) [15]. Statistical properties of this proposed loss function are presented in the Appendix. The discrepancy between the proposed and the exponential loss function is that the penalty associated with the proposed loss function is less than that of the exponential one, if the misclassification error rate at current stage is not equal to zero (shown in Fig. 6.3 where loss is plotted against a function of the classification margin $y.f$). This modification offers optimal weight on the misclassified samples at each boosting iteration and enforces to classify correctly in the next few subsequent iterations to the misclassified samples found at any iteration. Thus the proposed loss function encourages fast convergence of boosting iteration. One additional advantage of this proposed loss function is that the user can regulate the amount of penalty for negative margins after observing the classifier performance over a validation data set. Accordingly, we determine the value of λ through cross validation (λ is a function of k shown in the appendix and the value of k is determined experimentally). We derive a slightly modified Adaboost algorithm and we name it “Adaptive Regularized boosting (ARboost)” by minimizing the proposed loss function as shown in Table 6.1 (The derivation of minimizing the proposed loss function is shown in Appendix).

```
1. Initialize the observation weights ,
   $w_i^+ = \frac{1}{N^+},\ w_i^- = \frac{1}{N^-},\ i = 1, 2,..., N,\ N^+ + N^- = N$ where,
   '+' and '-' represents positive and negative samples respectively.
2. For $m$ = 1 to $M$:
   (a) Fit a classifier $G_m(x)$ to the training data using weights $w_i$.
   (b) Compute $err_m = \frac{\sum_{i=1}^{N} w_i I(y_i \neq G_m(x))}{\sum_{i=1}^{N} w_i}$
   (c) Compute
       $\alpha_m = \max\{\log k, \log(k(1-err_m)/err_m)\}, k > 0$
   (d) Set
       $w_i \leftarrow w_i \exp[\alpha_m I(y_i \neq G_m(x))], i = 1,2,...., N$
3. Output $f(x) = sign[\sum_{m=1}^{M} \alpha_m G_m(x)]$
```

Table 6.1: Proposed ARboost algorithm.

Proposed ARboost seeks the feature weight, $\alpha_m = \max\{\log k, \log(k(1-err_m)/err_m)\}, k > 0$ where, for the standard Adaboost [15] algorithm the value of k is always 1. This leads to the weights associated with misclassified observations at any stage to be k times as much as the existing Adaboost (derivation is shown in the appendix). The value of k for ARboost is determined by cross-validation and is discussed in the experimental section.

The additional term that we propose in the existing loss function acts as a regularizer in the boosting framework. There are two other well-known regularized boosting algorithms in the literature: ε –boosting [15] and l_1-regularized boosting [55]. The regularization strategy in ε-boosting uses the idea of shrinking the contribution of each feature (feature weight). ε-boosting is slower but easy to implement and shares some properties with l_1- regularized loss function. In l_1- regularized boosting, the exponential loss function is minimized with l_1- regularization. This provides sparse solution that enforces early stopping and acts as a regularizer. l_1- regularization is a well-known technique in feature selection since it provides sparse solution [15]. Our method can adaptively regulate the effects of regularization in the boosting framework by fine-tuning the value of k from the training set through cross validation. We will demonstrate a comparative study among ARboost, ε –boosting [15] and l_1-regularized boosting [55] in the next chapter (chapter 7).

Chapter 7

7. Experimental Results

We have carried out experiments on two challenging data sets: oil sand mining and leukocyte microscopy images. Oil sand mining images were captured as a part of the validation study of oil sand crusher design [25] and leukocyte images were captured as a part of inflammation study [11]. The practical relevance of these two studies is discussed below.

7.1 Validation of oil sand crusher design

Particular focus has been giving to reaseach and testing of a new type of oil sand slurry technology called rejectless or crushing less extraction by oil sand mining industry [34]. This new technology allows the bitumen slurry preparing process closer to the mine face and it allows for more extensive use of hydrotransport, which is a key enabling technology introduced some years ago [25]. The technology will allow shovels in mines to feed oil sand directly into mobile crushers and slurry preparation plants. Starting the slurry preparation process closer to the mine face moves oil sand more cheaply than trucks or conveyors, and it could also improve recovery rates and reduce equipment down-time [25]. Figure 7.1 illustrates rejectless oil sand mining operation.

Mining operation is the first step toward oil extraction from oil sand. Once the topsoil and vegetation is removed, shovels are brought in to dig, scoop overburden first and then oil sand, which is filled into large haul trucks. Oil sand lies buried 15 m or deeper below the overburden [51]. These haul trucks ferry oil sand to the remote crusher plant site in the traditional surface mining technology. Instead using haul trucks, shovel directly dumps oil sand into the feeding hopper of the crusher located in the vicinity of the mine face in this new mining operation as shown in Figure 7.1.

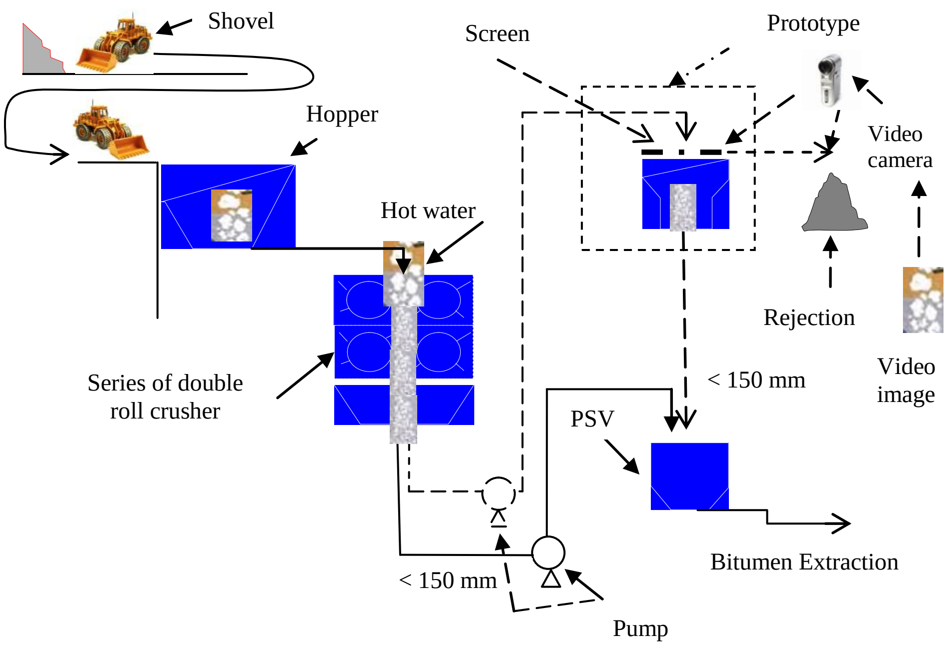


Figure 7.1: Schematic Diagram of reject less oil sand extraction process.

The crusher used in this mining operation consists of a series of double roll crusher. The aim of developing this technology is to use a new type of crushing system that will not generate any rejects and is therefore expensive vibrating screen is no longer needed to sieve unexpected crushed but bigger oil sand particles from rest of the smaller particles and to reject those bigger particles. Thus this technology will economize the oil sand production cost. Ongoing research on this technology envisages a mobile crusher where haul trucks will be eliminated from the system and the shovels can directly dump oil sand into the hopper of the crusher that can move around. This will entail into significant savings on operation and maintenance of haul trucks. This technology uses a wet crushing technology where the oil sand is mixed with hot water before feeding to the crusher. Flow of water helps to maintain a consistent flow of oil sand through the crusher and to break the wet oil sand particles into smaller ones. The output of the crusher should be oil sand particles of individual equivalent diameter less than 150 mm than can be directly transported to the Primary Separation Vessel (PSV). PSV separates oil sand, water and bitumen where oil sand is thrown into tailing ponds, water is recycled in the extraction system and bitumen is sent to the oil extraction plant for further processing which in turn produce crude oil.

With this reject less technology, more crushing of the ore will be done at the mine face. Without this technology, vibrating screens are used to separate large lumps from the ore feed and trucks then haul this material away. In this technology, there will be fewer conveyors and trucks required and there will be no screens to maintain. Thus this new technology has a much smaller environmental foot print as fewer trucks produce fewer emissions. Additionally, oil sand mining industry expects this technology to contribute to better recovery rates, as the large lumps that were previously rejected are now being broken down and the bitumen they contain recovered [6].

In order to develop this new technology a prototype of the new system was constructed in 2006 and was supplying bitumen for production by summer of 2007 [25]. But in order to be able to validate that this new crusher design would work, the size of the oil sand coming out of this crushing technology need to be measured, to make sure that these pieces were not too large (equivalent diameter of individual crushed pieces should be less than 150 mm), and would not cause blockages in the pumps that enforces crushed particles coming out from the crusher to send to the PSV. One way to validate the crusher design is to take samples of crushed oil sand coming out from the crusher and physically measure them with a ruler. This would require a lot of manual labor to measure a lot of dirt. To automatically verify the design of the crusher in the prototype of this technology, the output of the crusher is transported and is sieved through the screen. The undersize particles are sent to PSV and the oversize particles are rejected. Images of those rejected oil sand particles are acquired by the video camera mounted over conveyor belt located after the screen. These images are segmented by the automatic snake based segmentation technique [49] proposed in this research and the size of each oil sand particle is measured. The dotted components (screen) shown in Figure 7.1 were additionally used in the prototype to automatically validate the wet crushing technology and will no longer be used in the final oil sand production system.

So this new oil sand mining technology is a tremendous cost saving, up-to-the-minute technology developed at a minimal cost and handled hydrotransport in a lot more efficient way. The commercial-scale pilot of this new technology demonstrated positive results. The research continues and the technology is being vetted for future applications. Proposed snake based automatic segmentation technique is used to automatically validate the design of the crusher of this reject less wet crushing technology.

7.2 Importance of rolling leukocyte detection in the study of Inflammation

Leukocyte plays an important role in the study of inflammation. Inflammation is a natural defense mechanism initiated by tissue damage or injury, characterized by redness, heat, swelling, and pain. The key goal of inflammation is to restrict and eliminate the irritant and repair the surrounding tissue. Inflammation is an essential and useful process for the survival of the host. Three major stages are observed in the inflammatory response: first, capillaries dilate and hence blood flow increases; second, microvascular structural changes and plasma proteins escape from the bloodstream; and third, leukocytes transmigrate through endothelium and accrue at the site of injury [11]. During inflammatory response, endothelium cell is activated; leukocytes start deviating from mainstream blood flow and come in contact with activated endothelium cell. Then, leukocytes in contact with activated endothelium cell start moving slower than the mainstream blood velocity. This slow movement of leukocytes in contact with endothelium cell is called rolling. During rolling stages, leukocytes diffuse through vascular wall, reach the injured tissues and encounter the germs [38].

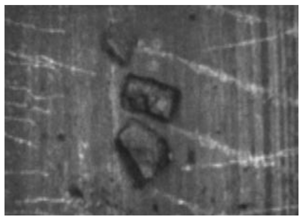
Although inflammation is a normal defense mechanism, it sometimes becomes harmful in various inflammatory diseases. Because sometimes the immune system attacks its host body due to an inability to distinguish invading organisms from the body’s own cells and causes inflammatory diseases. To struggle against such diseases, anti-inflammatory drugs are developed by blocking or controlling any of the necessary processes of inflammatory response. Because blocking any of the processes can severely reduce leukocyte accumulation at the injured site. The rolling velocity is a key predictor of inflammatory cell recruitment [38]. The most common description of leukocyte rolling velocities is a velocity distribution, preferably for hundreds of cells. Inflammation research involves the study of the velocity distribution of leukocytes. The roles of rolling velocities of leukocytes in acute and chronic inflammation are being investigated *in vitro* (“within the glass”, *i.e.* in a test tube) model systems and *in vivo* (“within the living”) microcirculation studies [11]. In this experiment, *in vivo* microcirculation studies have been considered where experiements have been conducted on living mice. To measure and analyze the leukocyte rolling velocity from the *in vivo* experiements, the movements of the leukocytes inside the postcapillary vennule of a cremaster muscle of a mouse are observed in video recordings made though a camera coupled with the intravital microscope. Typically such studies require tracking tens of thousands of leukocytes every day for clinical trials and drug discovery. To obtain the velocities of leukocyte or to initiate tracking of leukocytes, leukocytes are required to segment first. Normally, skilled technicians are employed to manually inspect hours of video and to extract the desired regions. Such manual processing is subject to operator errors and biases, extremely time consuming, tedious, has poor reproducibility and above all not cost effective.

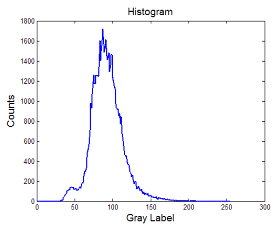
Proposed snake based automatic segmentation technique [48] can automatically segment leukocytes from leukocyte microscopy images captured during *in vivo* experiment.

Several Challenges involved in automatic delineation of oil sand particles from oil sand images and leukocytes from leukocyte microscopy images are discussed below.

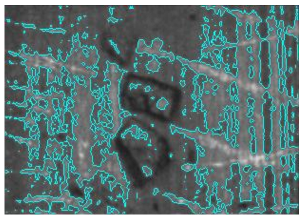
7.3 Challenges Involved in delineating oil sand particles

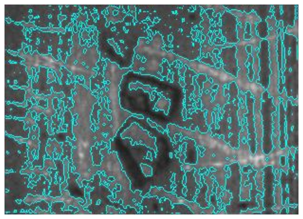
Figure 7.2(a) shows example of an oil sand image. Figure 7.2 (a) shows that the oil sand particles are relatively darker than its background. We apply some well-known state-of-the-art algorithms, namely, Otsu's global thresholding [35], Chan *et al.*'s [9] well-known locally adaptive variational thresholding, Chan and Vese's [1] region based level set method on oil sand particles to delineate oil sand particles from the rest of the image. The green lines in Figure 7.2(c), 7.2(d) and 7.2(e) are the object boundaries found by Otsu's global thresholding [35], Chan *et al.*'s [9] adaptive variational thresholding and *Chan and Vese*'s [1] level set method respectively. These results show that all of them fail to delineate oil sand particles from the conveyor belt as shown in Figure 7.2(c), 7.2(d) and 7.2(e) respectively. These poor performances of the existing of-the-shelf automatic segmentation techniques on oil sand images reveal difficulty of automated analysis. One of the key observations of these poor performances on automatic oil sand particle delineation is that these techniques attempt to classify pixels into background and foreground classes based on the gray level intensity differences between foreground and background classes and eventually these methods demonstrate poor performances since there is not much difference in gray level between oil sand particles (objects) and corresponding conveyor belt (background) pixels distribution (unimodal histogram shown in Fig. 7.2(b)). Besides, these images are often illuminated and noisy with various

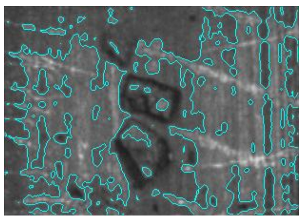

(a) Oil Sand image



(b) Histogram of Fig. (a)


(c) Otsu's Global thresholding


(d) Chan, Lam and Zhu


(e) Chan-Vese's algorithm

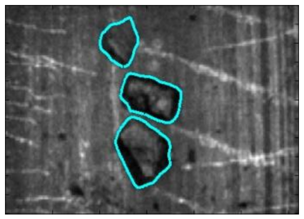

(f) GVF snake

Figure 7.2: Results of different well-known segmentation methods on oil sand image.

kinds of background (conveyor belt) clutter. Moreover, oil sand particles come in a variety of shape, size and texture. The apparent brightness of the individual object varies from object to object. Most of the times, objects are mixed with dirt and fine materials. Additionally, since the mine operates 24 hours a day, and the oil sand material needs to be analyzed outdoors, varying lighting and weather conditions play a significant role in their appearance in the image. The aforementioned factors constitute the main challenges to automatically segmenting the individual oil sand particles from these images. Figure 7.2(f) shows that Xu and Prince's [56] Gradient Vector Flow (GVF) can successfully delineate oil sand particles where the initial seeds are placed manually inside the oil sand particles. Unlike pixel label classification based automatic segmentation techniques, a snake energy functional is designed in such a way so that minimizing the energy functional provides forces to the manually drawn initial snake contour lock onto nearby features such as lines or edges and thus the snake captures the region (blob-object) surrounding the desired feature [21].

7.4 Challenges Involved in delineating leukocyte boundaries

Some potential challenges involved in delineating leukocytes as shown in Figure 7.3 are mentioned below [38]:

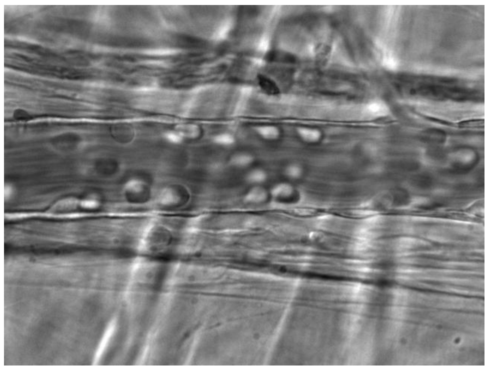


Figure 7.3: Leukocyte microscopy image.

- Background is moving in the leukocyte video because Leukocytes roll (slow movement) in a fast moving blood stream.
- Leukocytes slowly deform in shape as they roll along a venule.
- Breathing movements of the living subject causes jitter in the video.
- Image clutter presents in the images due to blood flow, venule wall, and presence of partially overlapping and osculating leukocytes.

- Image contrast changes due to change in transillumination conditions, breathing movement of the living subject.
- Leukocytes appear in dark or bright since the leukocytes go below/above the microscope focal plane.

7.5 Organization of the datasets

For oil sand mining images, we construct training set using 20 images, validation set using 30 images and test set using 100 images sampled randomly from an online video of oil sand particles over conveyor belt. For leukocyte images, we have carried out experiment on a training set of 5, validation set of 10 and a test set of 25 leukocyte images. Ground Truth of both of these datasets was generated by the experts before commencing the experiment. We have demonstrated the efficacy of the proposed snake initialization (PQT), evolution (IDGVF) and validation (ARboost) methods in this chapter (chapter 7).

7.6 Snake Initialization

We have conducted a comparative analysis among three automatic initialization techniques: Center of Divergence (COD) [13], Critical Point (CP) [54], Force Field Segmentation (FFS) [24], Poisson Inverse Gradient (PIG) and Probabilistic Quad Tree (PQT). Seeds generated by COD [13], CP [54] and ROIs found by proposed PQT method as well as initial contours found from white dots in Figure 7.4.

Table 7.1 illustrates the number of seeds or initial contours generated by the

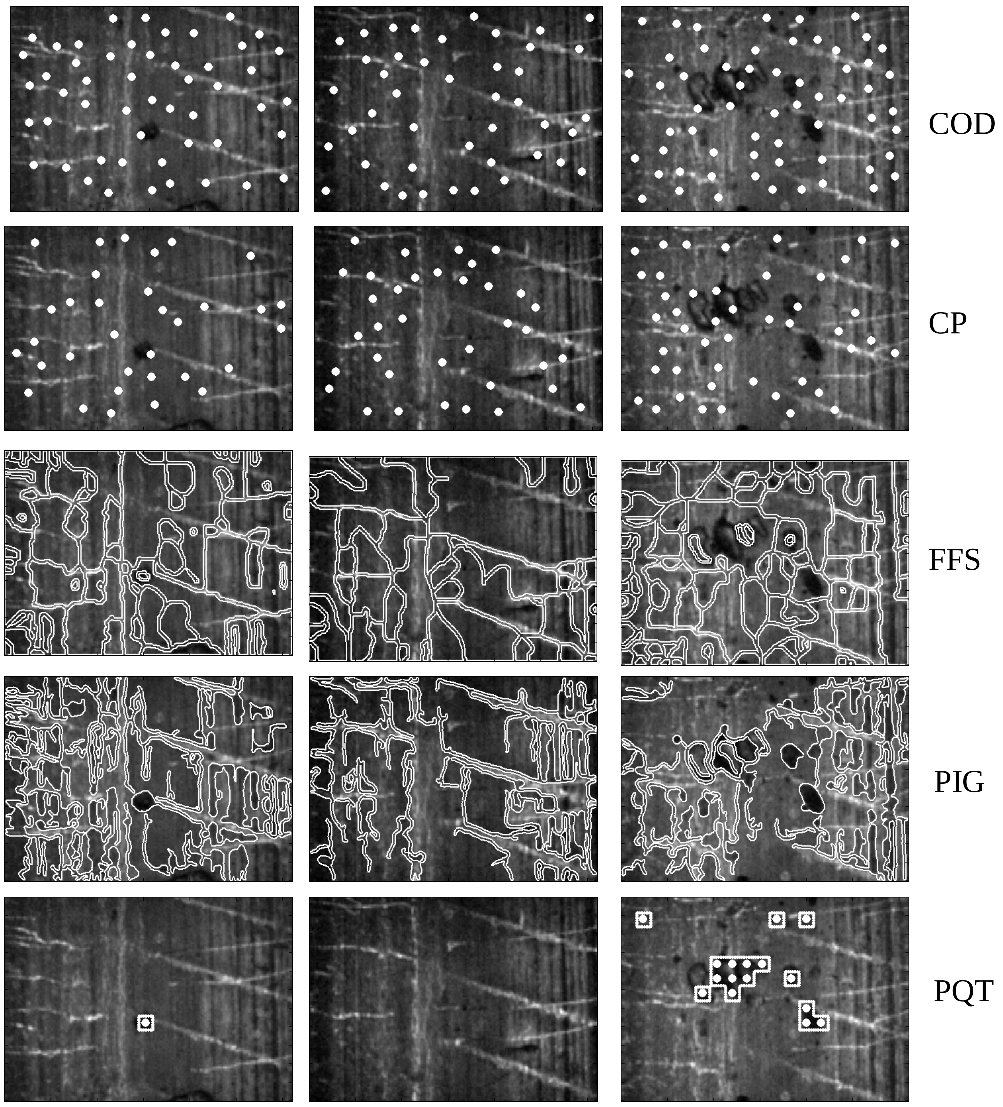


Figure 7.4: Results of different snake initialization methods.

COD, CP, FFS, PIG, and PQT. Figure 7.5(a) and 7.5(b) show the accuracy and F-measure for COD, CP, FFS, PIG, and PQT techniques. We have used IDGVF for snake evolution and ARboost for snake validation technique with all of these initialization methods to compute accuracy and F-measure. F-measure combines both recall and precision into a single value [41].

Datasets	*# of objects*	*# of Seeds generated by*				
		COD	*CP*	*FFS*	*PIG*	*Proposed* ***PQT***
Oil Sand	349	3215	2761	4757	3118	***686***
Leukocyte	193	1581	1104	2055	1250	***799***

Table 7.1: Comparison among existing snake initialization techniques.

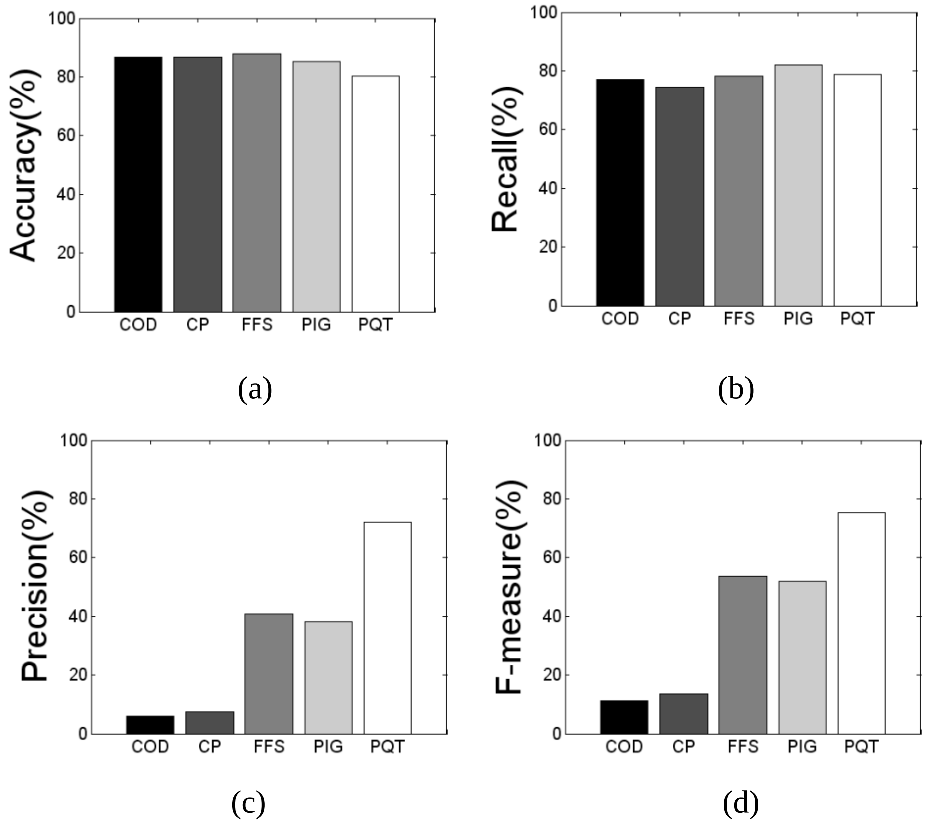


Figure 7.5: (a) Accuracy, (b) Recall, (c) Precision and (d) F-measure of different snake initialization methods on oil sand images.

Results show that although all techniques possess almost the same accuracy, PQT achieves the maximum F-measure value (25% more than that of PIG that achieves second highest F-measure value) and PQT generates significantly fewer seeds than the competing methods as shown in Table 7.1.

7. 7 Snake Evolution

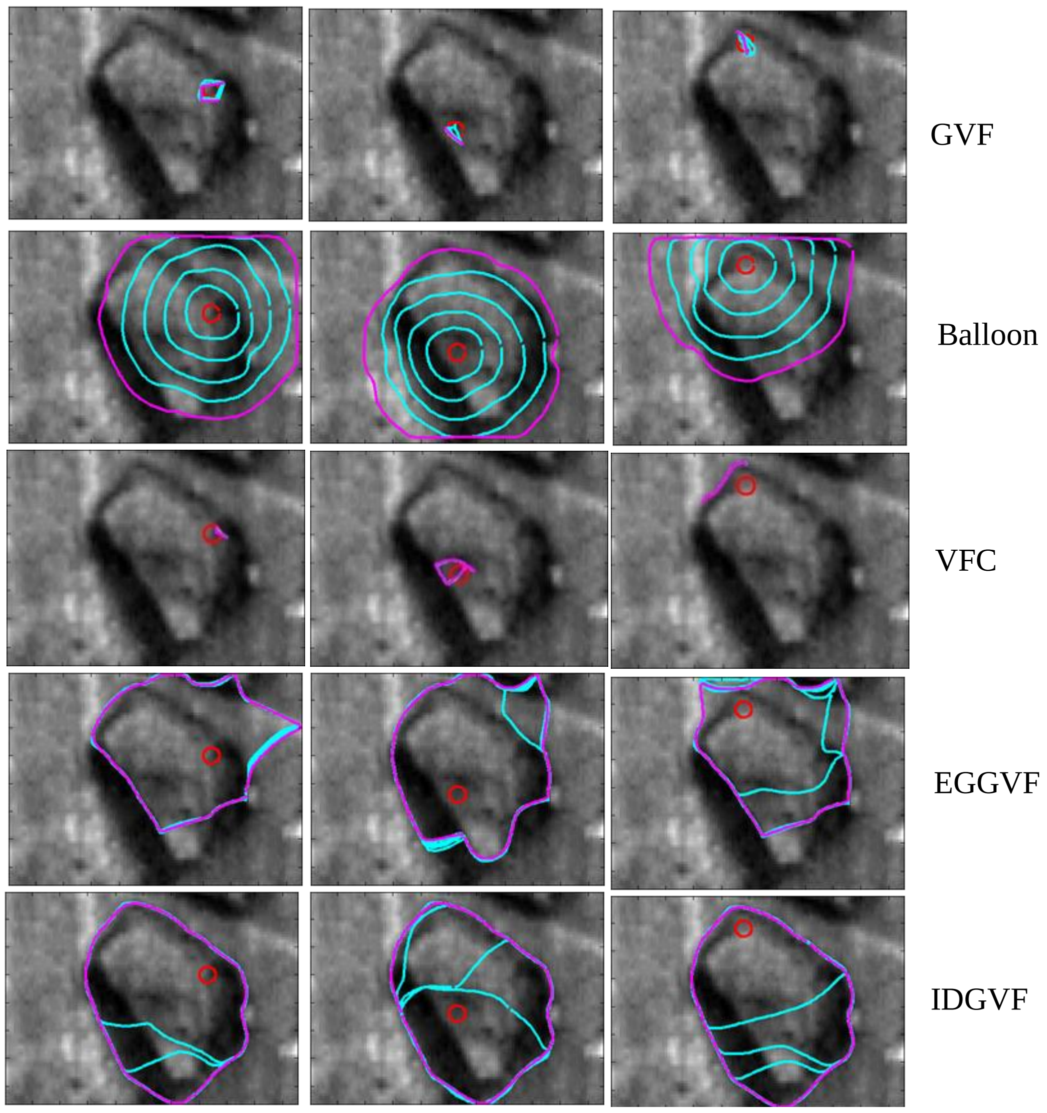


Figure 7.6: Results of different snake evolution algorithm from a small snake contour placed at three different positions within an oil sand particle.

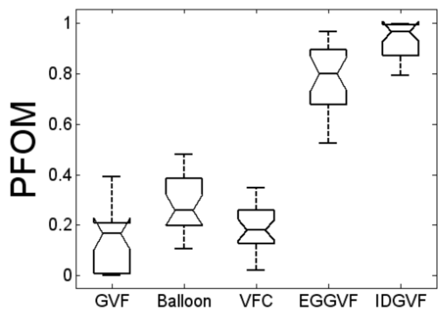
In the next set of experiments, we evaluate the efficacy of proposed IDGVF snake. We randomly choose 25 oil sand particles. In each one of these particles we place 5 randomly chosen seed locations. Then perform GVF [56], balloon snake [5], VFC [23], EGGVF [37] and proposed IDGVF snake evolution from these seed locations.

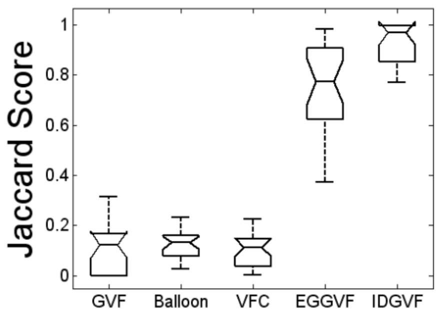
Fig. 7.6 illustrates results of GVF, Balloon snake, VFC, EGGVF and IDGVF snake evolution evolved from three different initial seed points chosen arbitrarily inside an oil sand particle. Both visual results and higher PFOMs and Jaccard Score values demonstrate that IDGVF is less sensitive to the choice of seed locations compared to other snake algorithms. It has a broad capture range and it can capture contour from a seed point located inside an oil sand. Dirichlet boundary condition enables a snake to grow from a naïve initial contour (circle of radius ~2-3 pixels) located within the blob-object and help to reach at actual object boundary. Integration of directional gradient information in the force vector field in an interleave fashion guides snake to surpass weak edges present inside the solid object due to image noise, and helps to sense the actual object edges. IDGVF is handy to capture contour of the blob like object.

To measure the performance quantitatively we compute Pratt's figure of merit (PFOM) [1] and Jaccard score [16], both of which is dimensionless number and is bounded by 0 and 1. PFOM is defined as:

$$PFOM = \frac{1}{\max(I_I, I_A)} \sum_{i=1}^{I_A} \frac{1}{1 + \beta d^2(i)} \tag{7.1}$$

Where, I_I and I_A are the number of ideal and actual edge pixels, $d(i)$ is the pixel miss distance of the i-th edge pixel detected, and β is a scaling constant chosen to be 1/9 to provide a relative penalty between smeared edges and isolated, but offset, edges [1]. PFOM measures the inverse of the square of the distance among actual edge pixels and edge pixels found by the algorithm [1]. Jaccard Score is the ratio of the intersection and union of the area of the objects found in an image by the algorithm and the area of the actual objects. The value of Pratt's figure of merit (PFOM) and Jaccard Score is bounded by 0 and 1. Superior performance of a segmentation algorithm is indicated by higher PFOM or higher Jaccard Score values. Figure 7.7 (a) and 7.7 (b) show box plot of the PFOMs and Jaccard Scores for GVF, balloon snake, VFC, EGGVF and IDGVF snake.





(a) (b)

Fig. 7.7: (a) and (b): Segmentation Scores (Pratt's Figure of Merit (PFOM) and Jaccard Score) for different snake evolution algorithms on

7. 8 Snake Validation

7. 8.1 Snake Validation by Principal Component Analysis

(a) Experiments on oil sand images:

We construct 10 rectangular pattern images by warping an annular ring of width 5 pixels both inside and outside across the contour of 10 oil sand particles and use them for PCA computation. Figure 6.3 demonstrates first 10 eigenvalues in descending order for the training set consists of 10 pattern images of oil sand particles. The Scree plot illustrated in Figure 6.3 shows that the first eigenvector corresponding to maximum eigenvalue is sufficient to compute PCA reconstruction error for snake validation since it represents >80% of the total variance [18]. Experiments show that only the first principal component linked to the maximum eigen value explains the maximum percentage of variance (80% of the total variance) according to the scree plot shown in Figure 7.8.

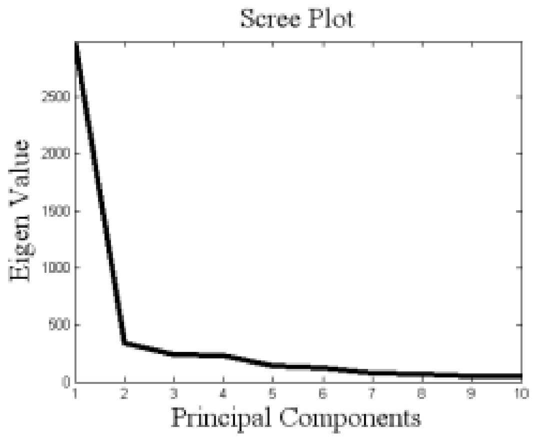
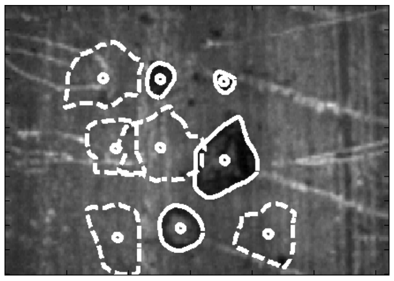
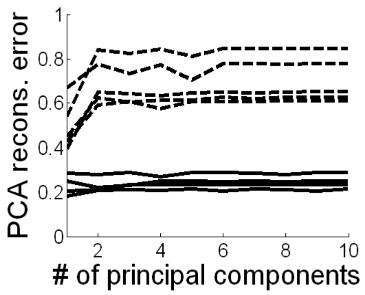


Figure 7.8: Scree Plot

Figure 7.9(b) also confirms this choice for a typical image that shows the first principal component has good discrimination capability.





(a) (b)

Figure 7.9: (a) Seed points (small circle) and evolved snakes. (b) PCA reconstruction errors of these snakes.

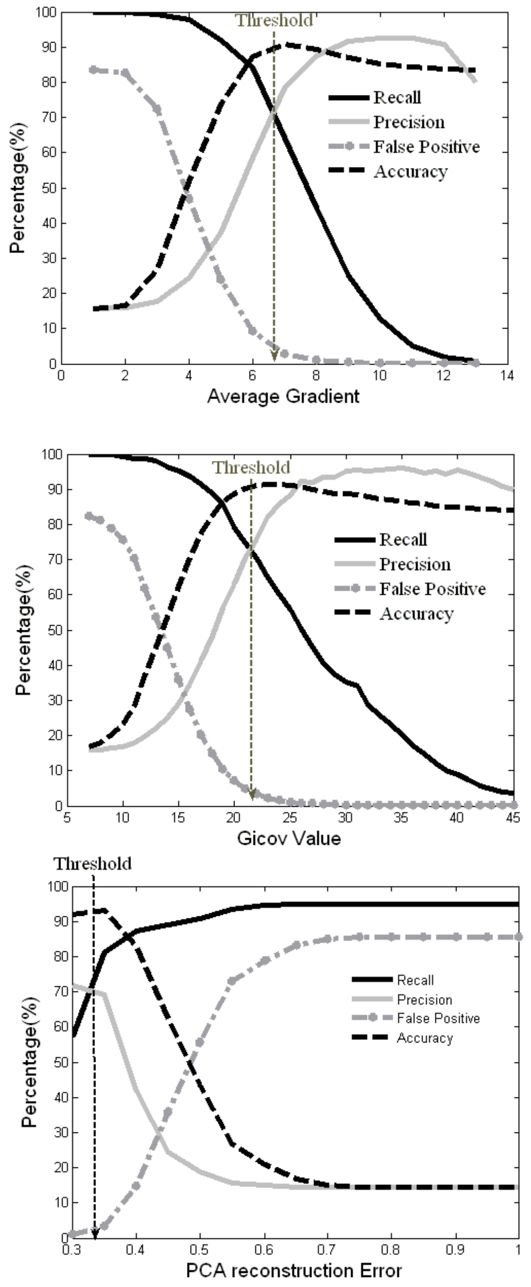


Figure 7.10: Threshold selection using cross validation (leave-one-out method)

We use cross validation (leave-one-out) [15] to compute the threshold for PCA reconstruction error shown in Figure 7.10. We first compute different performance curves *vs*. PCA reconstruction errors (see Figure 7.10). Then the threshold is chosen where the recall and precision curves intersect (Figure 7.10).

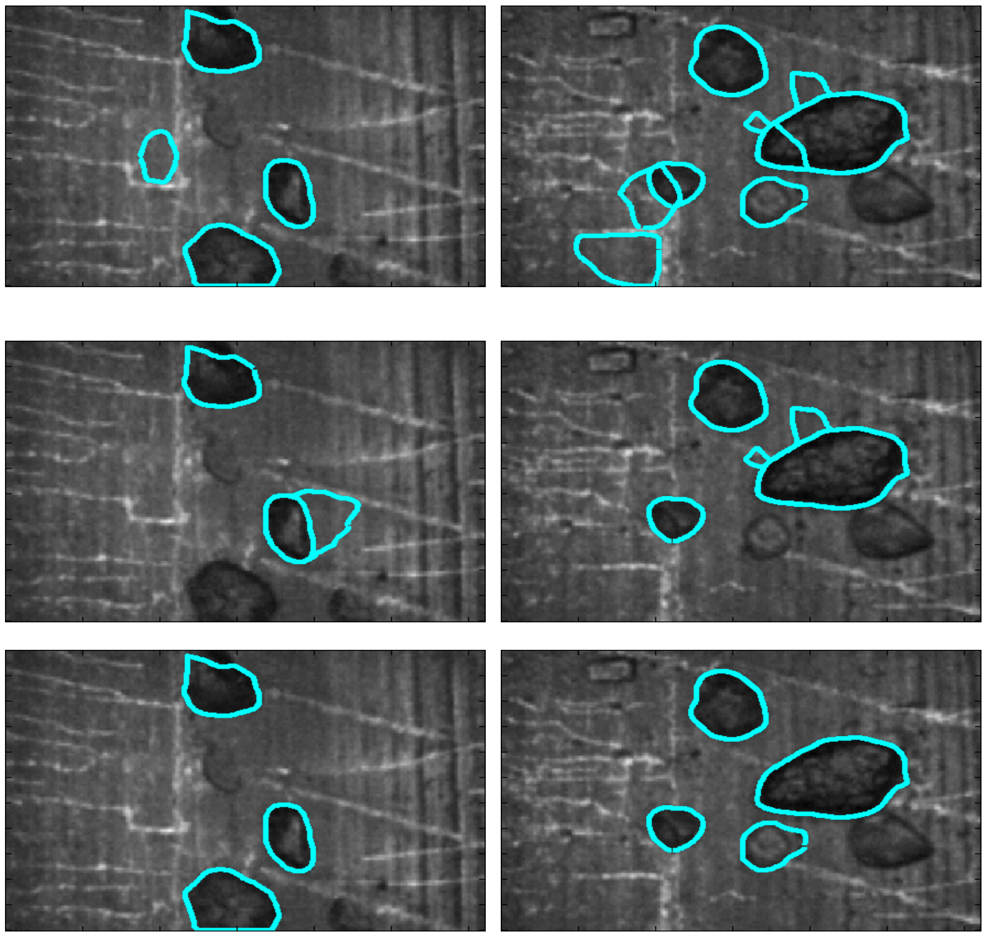


Figure 7.11: Oil sand images. Top row: results of average gradient. Middle row: results of GICOV. Bottom row: results of our proposed PCA algorithm.

We have compared our PCA based outlier detection with two gradient based Techniques: GICOV [10] and average directional image gradient computed on evolved snake contours. GICOV is defined as the ratio of average directional image derivative and their standard deviation across snake contour. Snake associated with GICOV or average gradient value greater than a predefined threshold considers as desired object, otherwise it is considered as a clutter. We compute these thresholds on the same training set used for PCA using cross validation (leave-one-out) technique. Once again the threshold is chosen using where the recall and precision curves intersect each other on the training image set (see Figure 7.10).

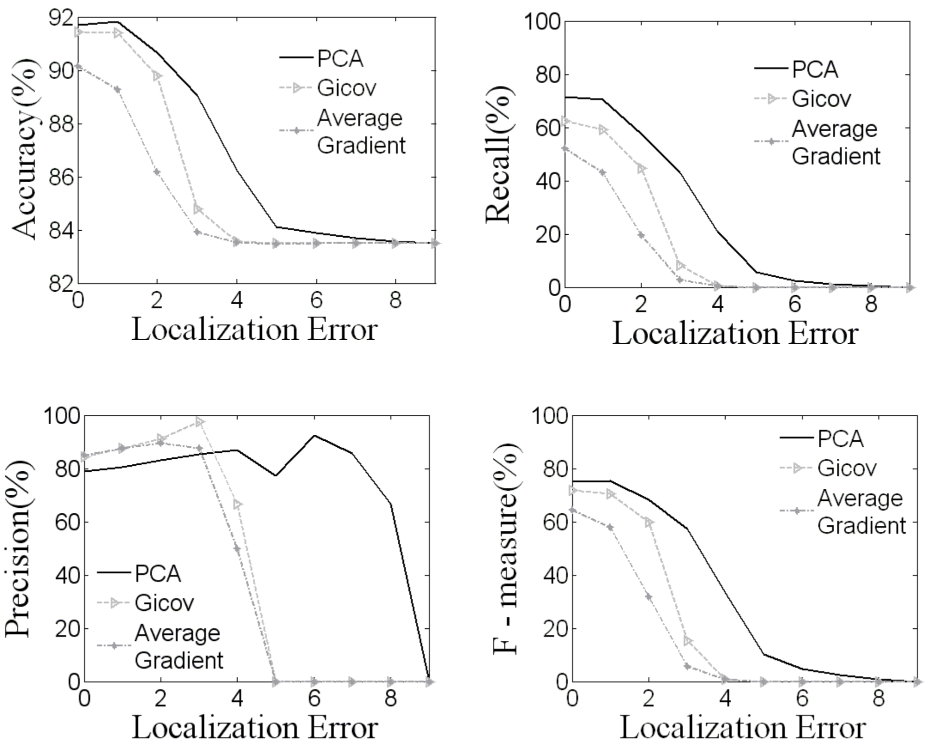


Figure 7.12: Accuracy, Recall, Precision and F- measure vs. localization error.

Detections obtained by the proposed PCA, GICOV and average directional derivative techniques are shown in Figure 7.11, which demonstrates that the proposed algorithm is superior to its competitor for the images at hand. To measure the robustness, we have randomly shifted each converged snake contour some pixels (up to 9 pixels) along horizontal and vertical direction and have measured performance of these three techniques. We define the total amount of shift of each converged snake contour from its local minima as localization error. We measure accuracy, recall, precision and F-measure for different localization errors (in pixels) shown in Figure 7.12. Figure 7.12 shows that PCA-based snake confirmation test is up to 10% more accurate, recall is up to 20% better and it can sustain more localization error (at least 5 to 6 pixels).

The F-measure value for proposed PCA technique is always larger than those of GICOV and average gradient. The robustness of the proposed PCA-based method can be attributed to the proposed annular pattern images that capture image texture information near the object boundaries. Thus our method can still be effective if the snakes converge slightly away from expected object boundaries due to noise and clutter that frequently plagues many snake-based tools.

(b) Experiments on leukocyte images

Automatic leukocyte detection from intravital microscopy images help in the study of inflammation as well as in the design of anti / pro – inflammatory drugs [10]. Detections obtained by the proposed PCA, GICOV and average directional derivative techniques are shown in Figure 7.13.

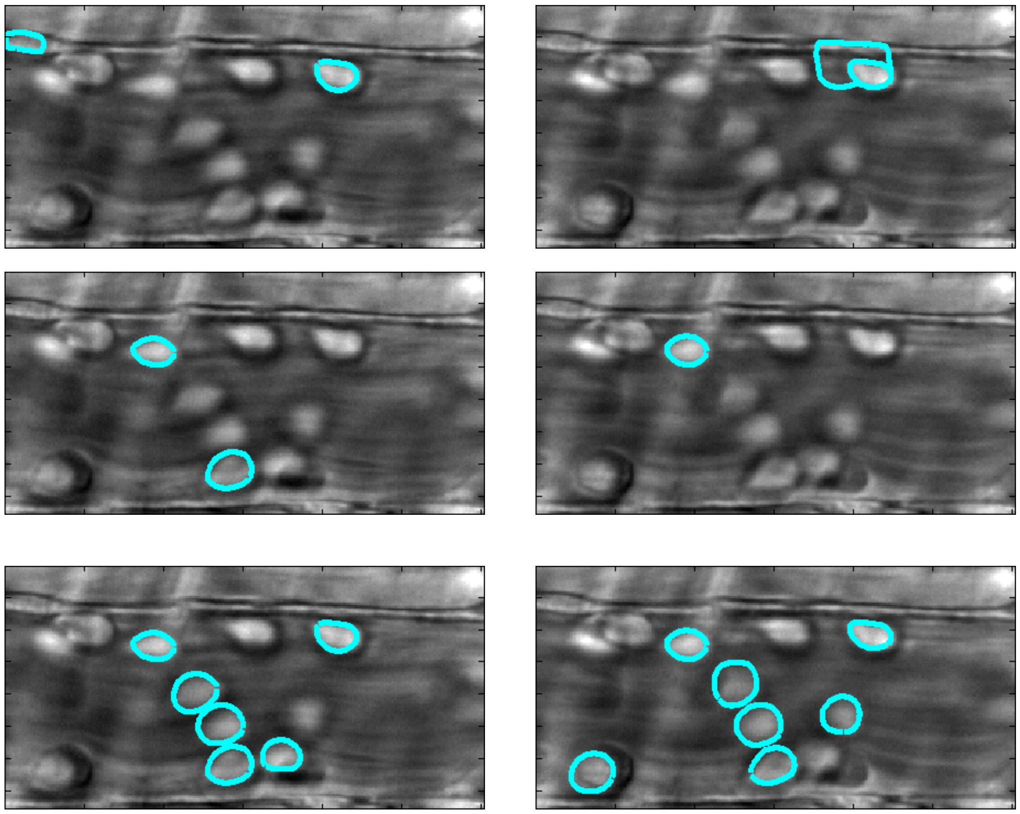


Figure 7.13: Leukocyte images. Top row: results of average gradient. Middle row: results of GICOV. Bottom row: results proposed algorithm.

Accuracy, recall, precision and F-measure for different localization errors (in pixels) are shown in Figure 7.14. These comparisons show that proposed PCA validation is up to 5% more accurate, recall is up to 30% better and it can sustain more localization errors (at least 5 to 6 pixels) than its competitors.

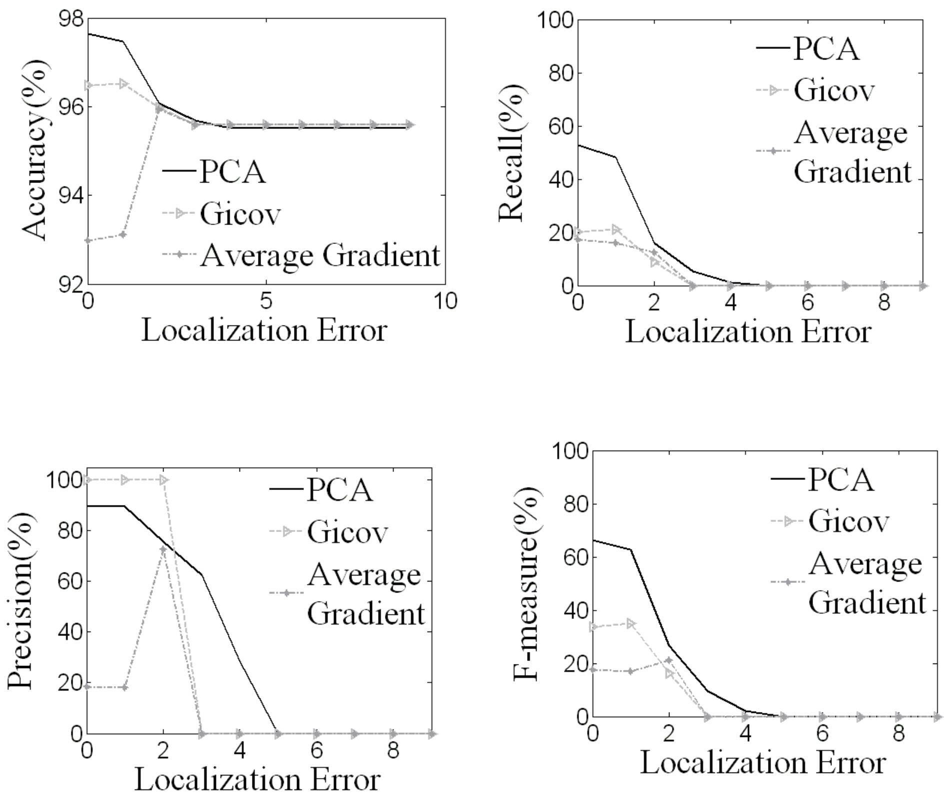


Figure 7.14: Accuracy, Recall, Precision and F- measure vs. localization error for leukocyte images.

7. 8.2 Snake Validation by Boosting

(a) Determination of the value of *k*

We determine the value of *k* (discussed regarding feature weight in chapter 6.2) using five-fold cross validation technique [15]. We compute misclassification errors for different values of *k* that are shown in Figure 7.15(a) on oil sand images. Standard error bars indicate the standard errors of the individual misclassification error rates for each of the five parts. It is observed that both the average misclassification error rate and standard error is minimum for *k* = 8 for oil sand images. For standard Adaboost [15] algorithm, the value of *k* is always 1. ARboost outperforms the standard Adaboost algorithm because ARboost can choose the best value of k for which the misclassification error is the minimum. The misclassification error rate for boosting with decision stumps [15], as a function of the number of iterations for $k = 8$ is shown in Figure 7.15(b).

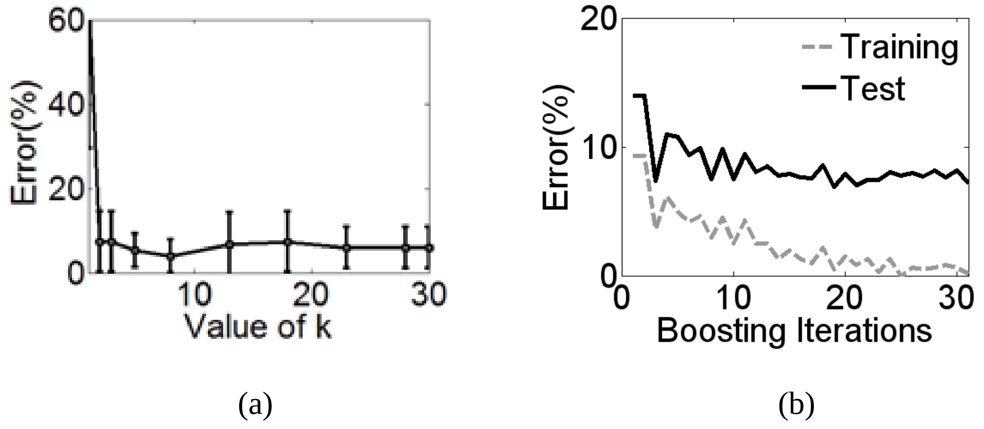
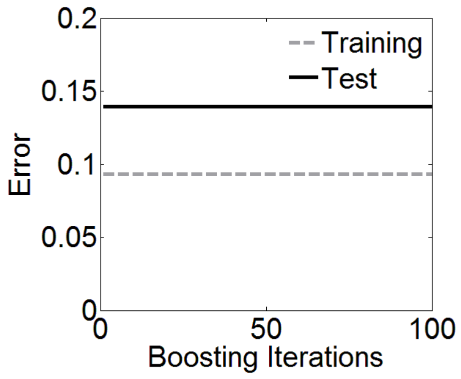


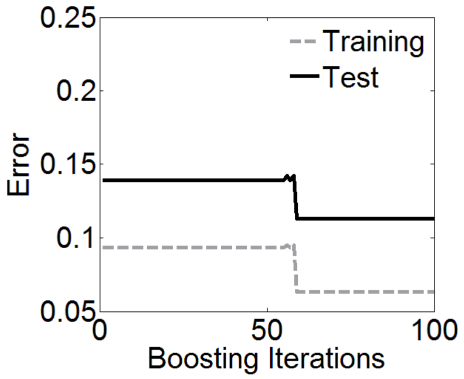
Figure 7.15: (a) fivefold cross validation curve with standard error bars; the curve has minima at $k = 8$. (b) Misclassification error rate over the number of iterations for oil sand images.

(b) Convergence of the Boosting Algorithm

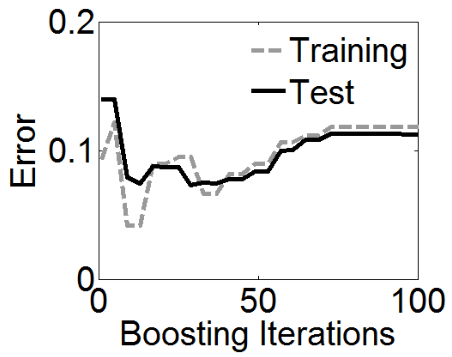
We empirically examine the performance of proposed regularization for the boosting framework and compare it with two other well known regularization techniques used in boosting: ε– boosting and l_1– regularized boosting. We have conducted experiments on oil sand images and the plot of misclassification error rates over boosting iterations on both training and test set for standard Adaboost, ε– boosting, l_1– regularized boosting and proposed ARboost is shown in Figure 7.16. This experimental result shows that the error rate does not change with boosting iterations for Adaboost. The reason behind this constant error rate could be inadequate weight adopted on misclassified samples at each



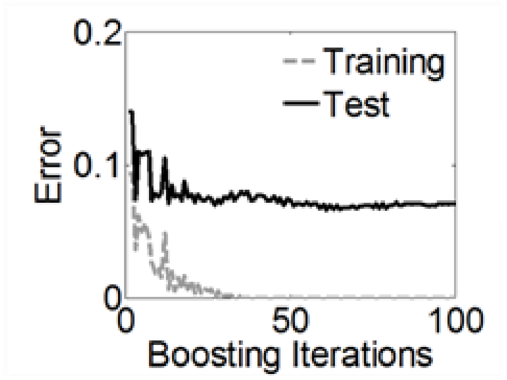
Standard Adaboost



ε - boost



l_1- regularized boost



Proposed ARboost

Figure 7.16: Error rates with number of iterations for different variants of boosting.

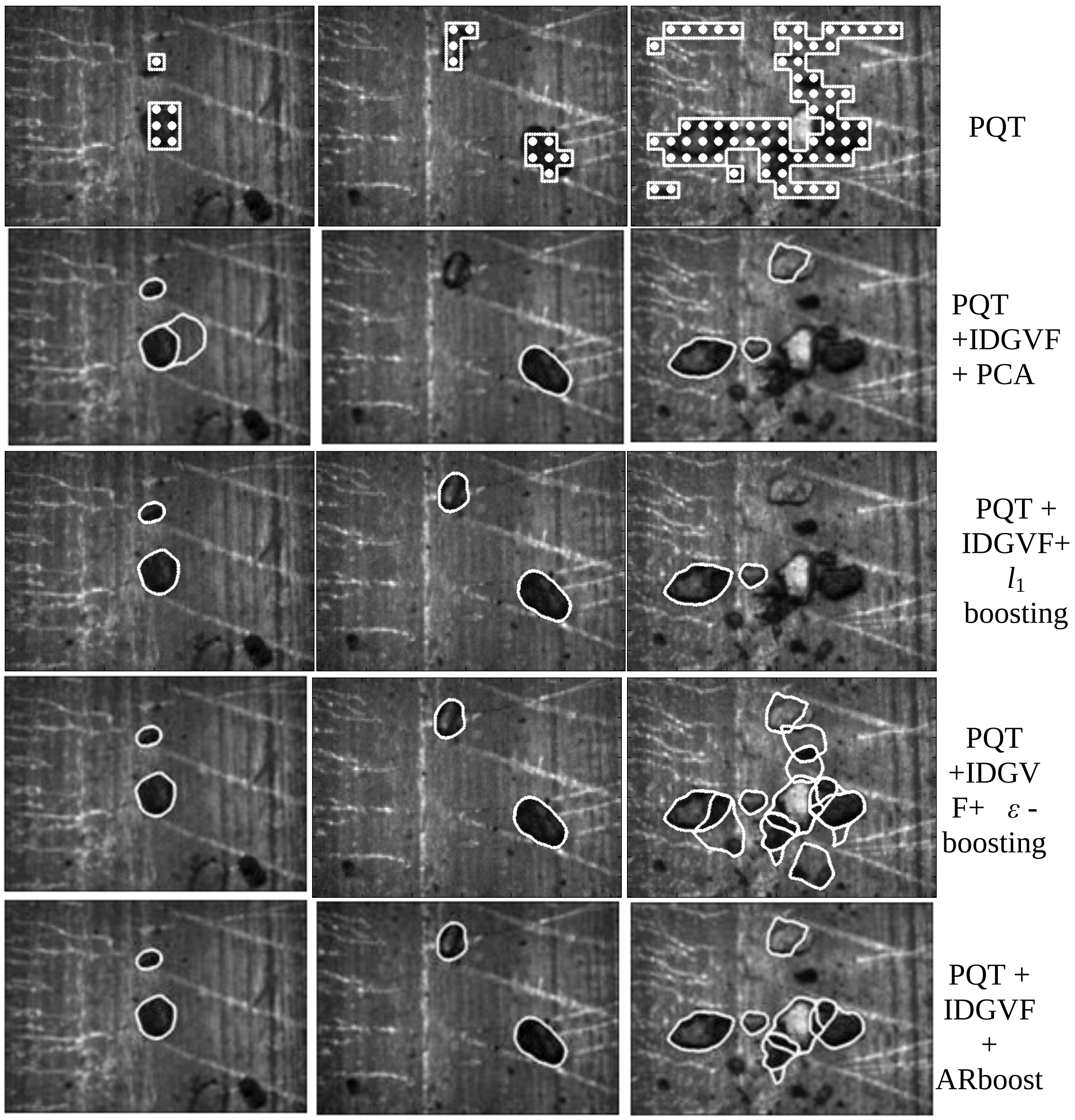

PQT
PQT +IDGVF + PCA
PQT + IDGVF+ l_1 boosting
PQT +IDGVF+ ε -boosting
PQT + IDGVF + ARboost

Figure 7.17: Comparison on different snake validation algorithm on Oil sand images.

iteration by Adaboost. Regularization in ε– boosting forces a decrease in the error rate; however, it cannot always maintain the monotonic decreasing trend of error rate over boosting iterations. Error rate in l_1– regularized boosting decreases as boosting iteration number increases; however, this decreasing trend of error stops after certain iterations and then the error rate starts increasing. Error rate in the proposed ARboost decreases as the boosting iteration number increases since ARboost determines adequate weight to the misclassified samples at each iteration and user can adopt the amount of regularization by fine-tuning the value of k through cross validation.

(c) Comparison with other variants of boosting

We have compared ARboost based snake validation technique with three other validation techniques: PCA [47], ε-boosting [15] and l_1- regularized boosting [55]. We have used PQT as snake initialization and IDGVF as snake evolution for all of these methods. The best values of the parameters of all techniques such as, threshold value of re-projection errors in PCA, the value of ε that controls regularization in ε-boosting have been selected using cross validation techniques. Detections obtained by all these methods on oil sand and leukocyte images are shown on Figure 7.17 and Figure 7.18 respectively. Quantitative comparisons in terms of Pratt's Figure of Merit (PFOM) and Jaccard Score in of these methods on oil sand and leukocyte datasets are demonstrated in Figure 7.19 and Figure 7.20 respectively.

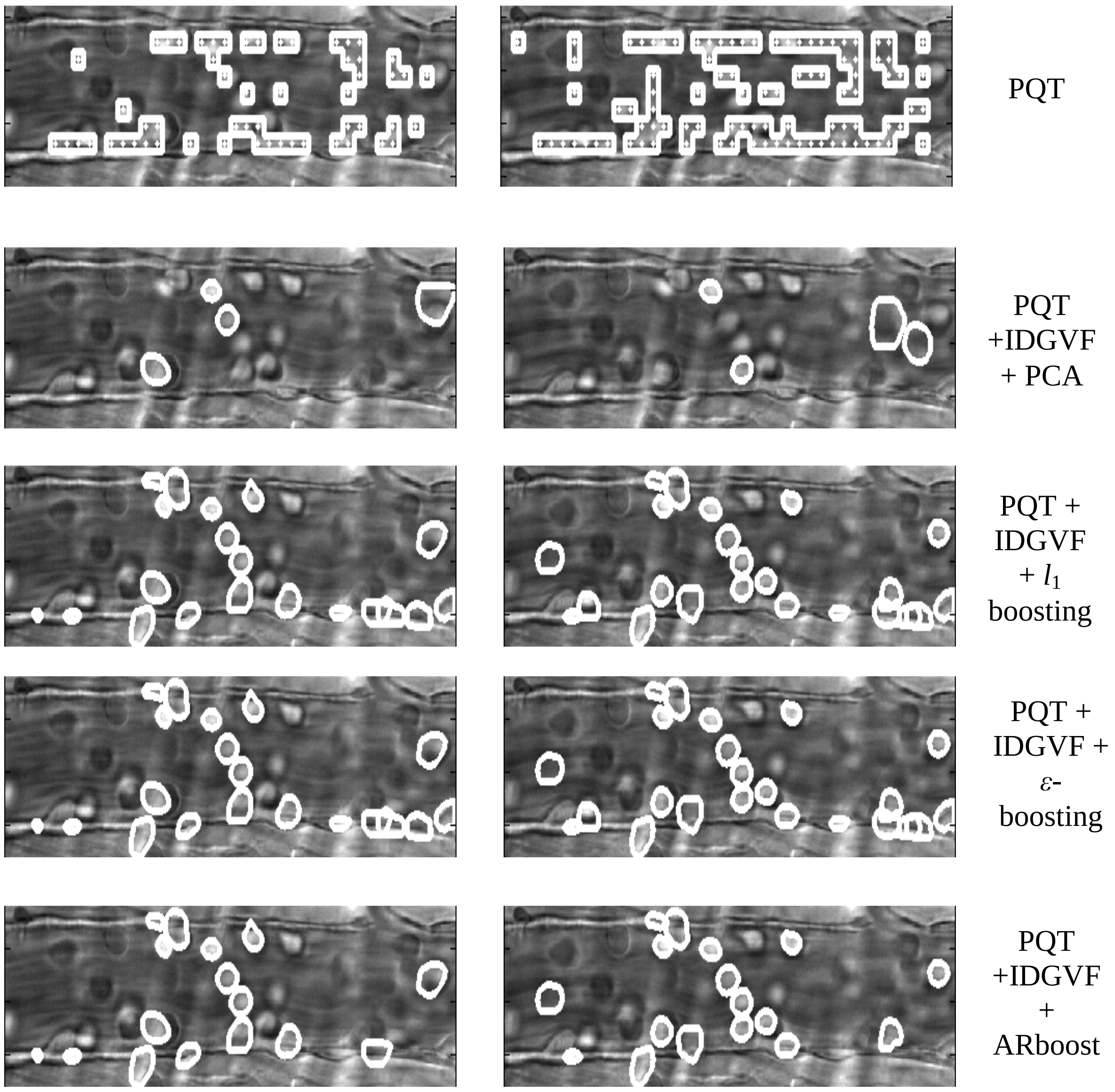

PQT
PQT +IDGVF + PCA
PQT + IDGVF + l_1 boosting
PQT + IDGVF + ε-boosting
PQT +IDGVF + ARboost

Figure 7.18: Comparison on different snake validation algorithm on Leukocyte images.

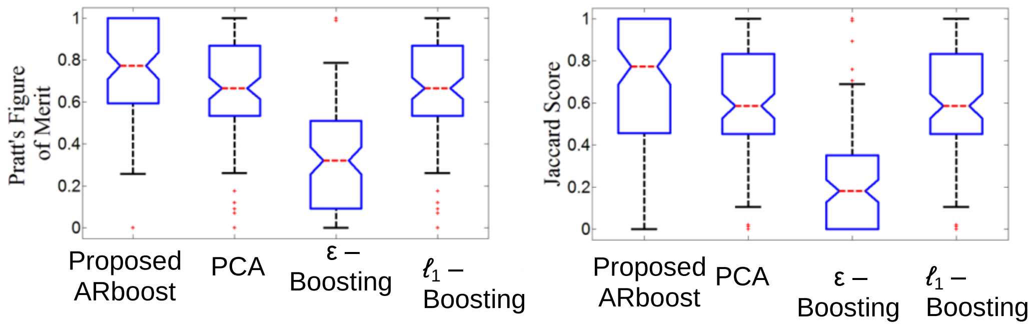


Figure 7.19: Segmentation Scores (Pratt's Figure of Merit (PFOM) and Jaccard Score) of different methods on Oil Sand images.

Both visual observations and quantitative results demonstrate that proposed ARboost based validation outperforms ε-boosting, l_1- regularized boosting and PCA since it can detect oil sand particles and leukocytes more accurately and precisely. Segmentation score (Jaccard Score and Pratt's Figure of Merit) as well as area under ROC curve of ARboost shown in Figure 7.21 (a) and 7.21(b) are greater than that of all other methods.

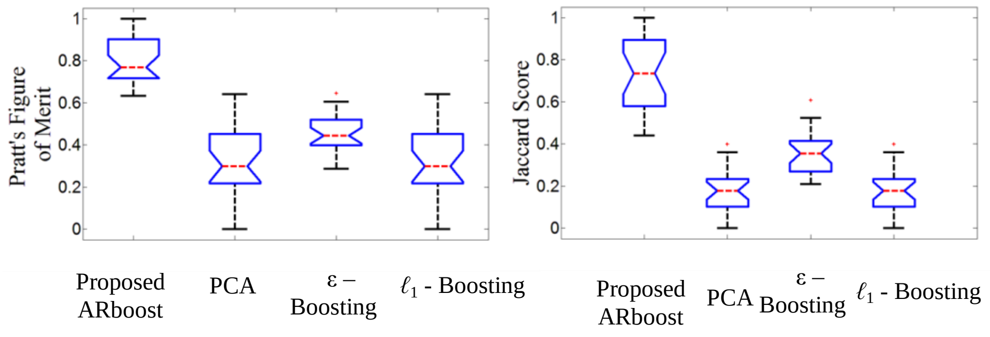
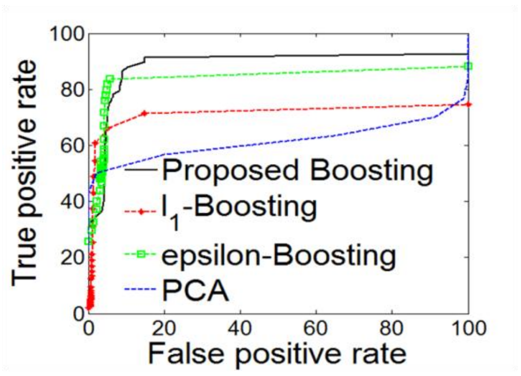
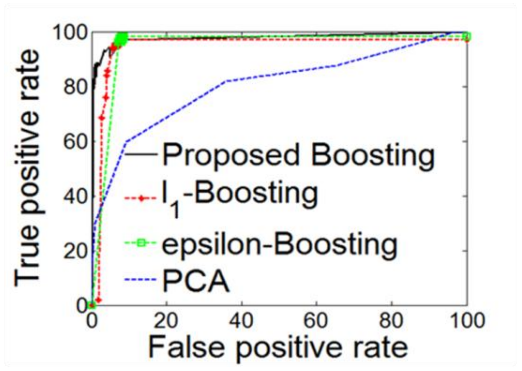


Figure 7.20: Segmentation Scores (Pratt's Figure of Merit (PFOM) and Jaccard Score) of different methods on Leukocyte images.





(a) (b)

Figure 7.21: Receiver Operating Characteristic (ROC) curves for (a) oil sand particles and (b) leukocyte images.

Chapter 8

8. Conclusions and Future Work

In this chapter (chapter 8) we have concluded the research work proving a summary and proposed new directions for future research.

8.1 Work Summary

(a) Automation of two emerging engineering applications

There are many real world applications that demand for automatic segmentation. In this thesis we have addressed two such emerging applications, oil sand particle delineation [59] and leukocyte detection [10] that requires complete automation. We have demonstrated that proposed active contour or snake based automatic segmentation technique can automatically segment oil sand particles and leukocytes.

(b) Snake automation by incorporating novel snake validation algorithm

Active contour or snake segmentation algorithm consists of two sequential steps: snake initialization and snake evolution. Success of snake based segmentation technique depends on careful placement of the initial contours. We have demonstrated that all of the available automatic initialization techniques lead to over-segmentation and consequently yield more than the required number of initial contours. Snake convergence is an ill–posed problem and it is very hard to impose all the constraints to ensure the convergence of the snake always onto desired object boundaries even snake initialization and evolution is performed well. This necessitates incorporating classifiers into automatic snake based segmentation framework that can separate evolved contours into object and non-object classes. Past endeavors concentrated only on the progression of snake initialization and evolution technique. In this research, we have first proposed in literature that snake based automatic segmentation technique consists of three sequential steps: snake initialization, snake evolution and snake validation. Though snake validation is a crucial step for snake automation, it has been practically ignored till date [47]. We place seeds uniformly within the regions of interest (ROI) of an image, evolve one Interleave Directional Gradient Vector Flow (IDGVF) snake from each seed and finally classify evolved snake contours into object and non-object classes.

(c) Probabilistic Quad Tree (PQT) based approximate segmentation technique for snake initialization

We have proposed a probabilistic quad tree (PQT)-based approximate image segmentation technique to automatically find ROIs within an image and then place seeds uniformly within ROIs. We have implemented a positional prior that facilitates a faster convergence of PQT than standard quad tree (QT) algorithm. Experimental results demonstrate that PQT generates fewer seeds than other existing snake initialization techniques without degrading the performance.

(d) Interleave Directional Gradient Vector Flow (IDGVF) for snake evolution

We have incorporated directional gradient information along with Dirichlet Boundary Condition (DBC) into Generalized Gradient Vector Flow (GGVF) evolution framework to capture oil sand particle or leukocyte boundary successfully from a small naïve initial contour placed inside the desired object. In addition, we compute snake evolution in an interleave fashion: evolve snake from initial naïve contour until convergence, then modify edge map based on the position of the contour converged at previous iteration and then evolve snake again until convergence. We continue this procedure iteratively until there is no considerable change in area enclosed by the snake in two successive iterations. We have demonstrated that this novel interleave computation guides snake to surmount weak edges present inside the object and to reach at object boundaries.

(e) Snake Validation by Principal Component Analysis (PCA)

We have intended Principal Component Analysis (PCA) based classifier to classify evolved snake contours into object and non-object classes. To perform PCA classification, we construct a novel pattern image by warping an annular band across each evolved snake contour and then converting into a rectangular image for computational convenience. This pattern image carries texture as well as dark to bright or bright to dark intensity transitions across the contour that characterizes oil sand particle or leukocyte contour. For a test image, snakes are evolved from given initialization, and then a pattern image is formed across each converged snake contour and then pattern image is formed into PCA space. The pattern image associated with lower PCA reconstruction error is recognized as oil sand particle or snake, otherwise it is considered as clutter. This PCA based classification is effective if prominent object boundaries are present.

(f) Snake Validation by boosting

We have also exploited the efficacy of multiple features (edge, region, shape etc.) using boosting over only edge based PCA classification. We offer a novel loss function for boosting regarding snake validation. We have incorporated one additional term in the existing exponential loss function for boosting that offers optimal weight on the misclassified samples in every boosting iteration that enforces to classify correctly the misclassified samples in the following iterations. These facilitate slightly fast convergence of boosting and derive adaptive regularized boosting (ARboost) by minimizing the proposed loss function. Results demonstrate that ARboost converges faster than other variants of boosting. Extensive experimental results demonstrate that proposed snake based automatic segmentation algorithm is computationally less expensive, more accurate and robust.

8.2 Future Work

We identify the following issues for future research with the evolution of snake towards automation for multiple blob-object detection:

- We would like to explore the efficacy of proposed loss function for boosting in multiclass classification problem.
- We would further like to investigate how the proposed loss function tackles the problem of over fitting.
- We would like to integrate snake validation step into snake evolution step that may facilitate early stopping of snake convergence.

- Since snake evolution is a computationally expensive step, instead of placing seed points uniformly within regions of interest (ROIs), we would like to further investigate seed placing method inside the ROIs.
- We would like to investigate the efficacy of incorporating other boundary condition such as, mixed boundary condition (different boundary conditions are used on different parts of the boundary) into IDGVF energy functional to facilitate snake more initialization independent as well as to sense weak edges located inside the object.
- We would like to automatically evaluate the weighting parameters in the active contour energy functional through cross validation.

Appendix

9.1 Supplements for Chapter 6

9.1.1 Features used for Snake Validation using Adaptive Regularized Boosting (ARboost)

(a) shape features [44]

$$\text{Formfactor} = \frac{4\pi\ \text{Area}}{\text{Perimeter}^2}$$

$$\text{Roundness} = \frac{4.\text{Area}}{\pi.\text{MaxDiameter}^2}$$

$$\text{AspectRatio} = \frac{\text{Max Diameter}}{\text{Min Diameter}}$$

$$\text{Elongation} = \frac{\text{Fiber Length}}{\text{Fiber Width}}$$

$$\text{Curl} = \frac{\text{Length}}{\text{Fiber Length}}$$

$$\text{Convexity} = \frac{\text{Convex Perimeter}}{\text{Perimeter}}$$

$$\text{Solidity} = \frac{\text{Area}}{\text{Convex Area}}$$

$$\text{Compactness} = \frac{\sqrt{\left(\frac{4}{\pi}\right)\text{Area}}}{\text{Max Diameter}}$$

$$\text{Modificati onRatio} = \frac{\text{Inscribed Diameter}}{\text{Maximum Diameter}}$$

$$\text{Extent} = \frac{\text{NetArea}}{\text{Area of the Bounding Rectangle}}$$

(b) Edge features

GICOV [10]: GICOV is defined as the ratio of average directional image derivative and their standard deviation across snake contour.

Let (X_i, Y_i), i=1:n be points on the snake contour. Let **n**(X_i, Y_i) denote the normal to the contour at (X_i, Y_i). Further, let I denote the image. Then gradient inverse coefficient of variation (GICOV) is defined as:

$$\text{GICOV} = \frac{m}{s/\sqrt{n}}, \quad \text{where,}$$

$$m = \frac{1}{n}\sum_{i=1}^{n} \nabla I(X_i, Y_i)\mathbf{.n}(X_i, Y_i), \text{ and}$$

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n} (\nabla I(X_i, Y_i)\mathbf{.n}(X_i, Y_i) - m)^2$$

Average gradient strength [47]: Average gradient strength is defined as the average directional image derivative. Let (X_i, Y_i), i=1:n be points on the snake contour. Let **n**(X_i, Y_i) denote the normal to the contour at (X_i, Y_i). Further, let I denote the image. Then average gradient strength (AGS) is defined as:

$$AVG = \frac{1}{n}\sum_{i=1}^{n} \nabla I(X_i, Y_i)\mathbf{.n}(X_i, Y_i)$$

Smoothness [60]: Smoothness of the snake contour is defined as,

$$\text{Smoothness} = \sum_{i=1}^{n-1} (\arctan E_{i+1} - \arctan E_i)^2$$

Where E_i and $E_{\text{i+1}}$ are two consecutive points on the contour, arctan E_i and arctan $E_{\text{i+1}}$ are their arctangent values and n is the total number of the points on the contour.

Monotonicity [60]: Monotonicity of a snake contour is defined as:

$$\text{Monotonicity} = \frac{1}{n}\sum_{i=1}^{n-1} \{1 \mid (\arctan E_{i+1} - \arctan E_i) > 0\}$$

Where E_i and E_{i+1} are two consecutive points on the contour, arctan E_i and arctan E_{i+1} are their arctangent values and n is the total number of the points on the contour.

Variance of the edge direction along the contour [60]: Variance value of the edge direction along the contour is defined as:

$$\text{V}ariance = \frac{1}{n}\sum_{i=1}^{n}(\arctan E_i - M)$$

Where E_i is a point on the contour, arctan E_i is the arctangent value of point E_i, n is the total number of points on the contour, and M is the mean arctangent value of all points on the contour.

Difference of Gaussian [14]: Difference of Gaussian (DOG) is used to detect edges. Image $I(x, y)$ is smoothed by convolution with difference of two Gaussians of with σ_1 and σ_2, *i.e*. DOG * $I(x, y)$ where,

$$DOG = \frac{1}{\sqrt{2\pi}}\left[\frac{1}{\sigma_1}e^{-(x^2+y^2)/2\sigma_1^2} - \frac{1}{\sigma_2}e^{-(x^2+y^2)/2\sigma_2^2}\right]$$

and (x, y) is the pixel location.

Residual error of the spin image [19] :

Spin images represents the global properties of any surface in an object-oriented coordinate system rather than in a viewer-oriented coordinate system. Object-oriented coordinate system is independent of viewer position.

Oriented points are used to generate spin images [19]. Oriented points (denoted as O) are 3-D surface points that have a direction [53]. The surface corresponding these points is represented as a polygonal mesh M with vertices. An oriented point O at a surface mesh vertex is defined by the 3-D position of the surface vertex (denoted as p) and a surface normal (denoted as n) [19]. A 2-D basis (p, n) that correspond to a local coordinate system can be formulated. The tangent plane *P* through *p* oriented perpendicularly to *n* and the line *L* through *p* parallel to *n* is used to achieve this goal. This leads to (α, β) cylindrical coordinate system where α is the perpendicular distance to while β is the signed perpendicular distance to *P*. Figure 1 illustrates cylindrical coordinate system.

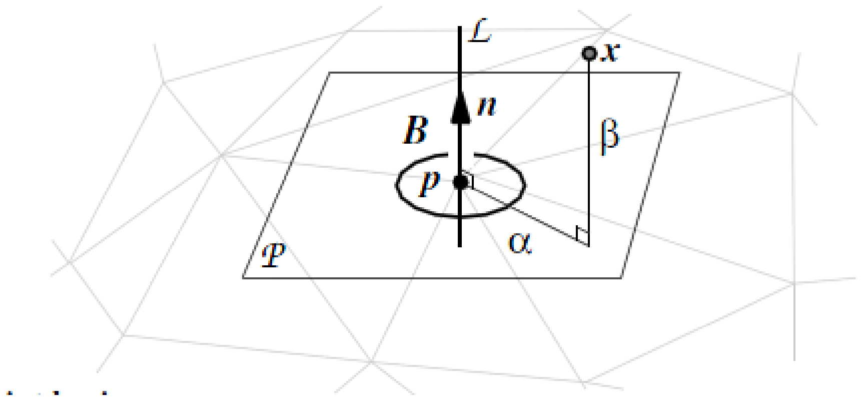


Figure 9.1: The cylindrical coordinate system and its (p, n) 2-D basis (taken from [19]) computed across the contour.

a spin map, S_o is generated using the oriented point basis. A spin map, S_o can be characterized as a projection function of the 3-D points *x* of an object to 2-D coordinates (α, β) associated with the 2-D basis (*p*, *n*) that corresponds to the oriented point *O*. The projection function is [2]:

$$S_o : \Re^3 \to \Re^2$$
$$S_0(x) \to (\alpha, \beta) = (\sqrt{\|x-p\|^2 - (n.(x-p))^2}, n.(x-p))$$

Although α cannot be negative, β can be both positive and negative.

Points are sampled uniformly across snake contour and a spin image vector is computed for each point. Then these points are learned and the residual error is computed similarly as PCA reconstruction error explained in section 6.1.2.

Edge lifetime [43]: Edge lifetime is defined as,

$$\text{Edge life time} = \frac{1}{n}\sum_{k=1}^{l}\sum_{i=1}^{n}\left|\nabla E_{i,\sigma_k}\right|$$

Where $\left|\nabla E_{i,\sigma_k}\right|$ is the absolute gradient value of the of the *i*-th point on the contour at scale σ_k, *n* is the total number of points on the contour.

Edge lifetime entails blurring the image over a range of scales, detecting edges at each scale, and tracking them across scale. The longer an edge persists before extinction (i.e. its lifetime) the more likely it is to be significant.

PCA reconstruction error [47]: has been defined in section 6.1.2.

Temperatue of a contour [7]: Let, P and Q be the length of snake contour and convex hull inscribes by the snake contour, then temperature (*T*) is defined as,

$$T = (\log(2 * P/(P-Q)))^{-1}$$

(c) Region based features

Intraclass variance (IV): let *D* be the vector consists of gray level values of the pixels located inside the bounding box that encloses the snake contour and *C* be the vector consists of gray level values of the pixels located inside the snake contour. Then IV = variance(C) + variance (D\C).

Interclass variance (INV): let *D* be the vector consists of gray level values of the pixels located inside the bounding box that encloses the snake contour and *C* be the vector consists of gray level values of the pixels located inside the snake contour. Then INV = variance (D) – IV = variance (D) - variance(C) - variance (D\C).

Region Instability [43]: let *D* be the vector consists of gray level values of the pixels located inside the bounding box that encloses the snake contour and *C* be the vector consists of gray level values of the pixels located inside the snake contour. Then instability is defined as the Area of the overlap between the histogram of C and D\C.

9.1.2 Derivation of Proposed Discrete Adaboost Algorithm

Proposed loss function is:

$$L(y, f(x)) = \exp(-yf(x) + \lambda \mid y - G(x) \mid), \lambda < 0.$$

Let $f_m(x) = f_{m-1}(x) + \beta_m G_m(x)$ be the strong classifier composed of first m classifiers. We can pose m-th iteration of Adaboost as the following optimization,

$$(\beta_m, G_m) = \underset{\beta,G}{\arg\min} \sum_{i=1}^{N} \exp[-y_i(f_{m-1}(x_i) + \beta G(x_i)) + \lambda \mid y_i - G(x_i) \mid]$$
$$\Rightarrow (\beta_m, G_m) = \underset{\beta,G}{\arg\min} \sum_{i=1}^{N} w_i^m \exp[-y_i \beta G(x_i) + \lambda \mid y_i - G(x_i) \mid] \text{ where,}$$

$w_i^m = \exp(-y_i f_{m-1}(x_i))$ is free of both β and $G(x)$.

$$\Rightarrow (\beta_m, G_m) = \underset{\beta,G}{\arg\min} [\exp(-\beta) \sum_{i:y_i = G(x_i)} w_i^m + \exp(\beta + 2\lambda) \sum_{i:y_i \neq G(x_i)} w_i^m]$$
$$= \underset{\beta,G}{\arg\min} [(\exp(\beta + 2\lambda) - \exp(-\beta)) \sum_{i:y_i \neq G(x_i)} w_i^m + \exp(-\beta) \sum_{i=1}^{N} w_i^m].$$

The solution for β_m and G_m can be obtained in two steps. First, for any value of $\beta+\lambda > 0$, the solution for G_m is:

$$G_m = \underset{G}{\arg\min} \sum_{i=1}^{N} w_i^m I(y_i \neq G(x_i)) / \sum_{i=1}^{N} w_i^m.$$

Let $err_m = \sum_{i=1}^{N} w_i^m I(y_i \neq G(x_i)) / \sum_{i=1}^{N} w_i^m$, then

$$\beta_m = \frac{\partial}{\partial \beta}(\sum_{i=1}^{N} w_i^m ((\exp(\beta + 2\lambda) - \exp(-\beta)) err_m + \exp(-\beta))) = 0$$
$$\Rightarrow \beta_m = \frac{1}{2}(\log \frac{1 - err_m}{err_m}) - \lambda = \frac{1}{2}(\log k \frac{1 - err_m}{err_m}), \text{ where, } \lambda = -0.5 \log(k),\ k > 0.$$

Also, $\beta_m + \lambda > 0 \Rightarrow \beta_m = \max\{\frac{1}{2}(\log k), \frac{1}{2}(\log k\frac{1-err_m}{err_m})\}$.

Now, $w_i^{m+1} = w_i^m \exp(-\beta_m y_i G_m(x_i))$.

Using the fact that $-y_i G_m(x_i) = 2I(y_i \neq G(x_i)) - 1$, we get,

$w_i^{m+1} = w_i^m \exp(\alpha_m I(y_i \neq G(x_i)))\exp(-\beta_m)$ Where,

$\alpha_m = 2\beta_m = \max\{\log k, \log(k/((1-err_m)/err_m))\}, k > 0.$

So, $w_i^{m+1} = w_i^m \exp(\alpha_m I(y_i \neq G(x_i)))$. The factor $\exp(-\beta_m)$ multiplies all weights by the same value, so it has no effect.

9.1.3 Statistical properties of the proposed loss function used in the Adaptive Regularized Boosting (ARboost)

Proposed loss function $L(y, f(x)) = \exp(-yf(x) + \lambda\,|\,y - G(x)\,|), \lambda < 0$ estimates one-half the log-odds of P (Y = 1|x). This justifies using its sign as the classification rule in $f(x) = sign(\sum_{m=1}^{M} \alpha_m G_m(x))$,This proposed loss criterion shows similar behavior with two other loss criteria with the same population minimizer, exponential loss criteria and binomial negative log-likelihood or deviance (also known as cross-entropy) [15]. It encodes that the function $f(x)$ that minimizes the L_2 version of the exponential criterion is the symmetric logistic transform of $P(y = 1|x)$. $E(L(y, f(x)))$ is minimized at $f(x) = \frac{1}{2}\log\frac{P(y=1\,|\,x)}{P(y=-1\,|\,x)}$. Here E represents expectation. Hence

$$P(y=1\,|\,x) = \frac{e^{f(x)}}{e^{-f(x)} + e^{f(x)}}$$

$$P(y=-1\,|\,x)=\frac{e^{-f(x)}}{e^{-f(x)}+e^{f(x)}}$$

Proof: Propose loss function can be written as,

$$\begin{aligned} L(y, f(x)) &= \exp(-yf(x)), \quad y=G(x) \\ &= \exp(-yf(x)+2\lambda), \quad \lambda<0 \text{ and } y\neq G(x) \end{aligned}$$

While E represents expectation over the joint distribution of *y* and *x*, it is sufficient to minimize the criterion conditional on *x*.

Case I: Consider $y=G(x)$

$$E(L(y, f(x))) = P(y=1\,|\,x)\exp(-f(x)) + P(y=-1\,|\,x)\exp(f(x))$$

$$\frac{\partial E(L(y, f(x)))}{\partial f(x)} = -P(y=1\,|\,x)\exp(-f(x)) + P(y=-1\,|\,x)\exp(f(x))$$

Setting $\frac{\partial E(L(y, f(x)))}{\partial f(x)}=0$ we have,

$$f(x)=\frac{1}{2}\log\frac{P(y=1\,|\,x)}{P(y=-1\,|\,x)}.$$

Case II: Consider $y\neq G(x)$

$$E(L(y, f(x))) = P(y=1\,|\,x)\exp(-f(x)+2\lambda) + P(y=-1\,|\,x)\exp(f(x)+2\lambda)$$

$$\frac{\partial E(L(y, f(x)))}{\partial f(x)} = -P(y=1\,|\,x)\exp(-f(x)+2\lambda) + P(y=-1\,|\,x)\exp(f(x)+2\lambda)$$

Setting $\frac{\partial E(L(y, f(x)))}{\partial f(x)}=0$ we have,

$$f(x)=\frac{1}{2}\log\frac{P(y=1\,|\,x)}{P(y=-1\,|\,x)}.$$

Bibliography

[1] I. E. Abdou and W. K. Pratt, “Quantitative design and evaluation of enhancement/ thresholding edge detectors,” *proceedings of the IEEE*, vol. 67, No. 5, pp. 753-763, may 1979.

[2] J. Besag, “On the Statistical Analysis of Dirty Pictures,” *Journal of the Royal Statistical Society*, *series B* (*Methodological*), vol. 48, no. 3, pp. 259 – 302, 1986.

[3] Y. Y. Boykov and M. P. Jolly, “ Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images,” *ICCV*, 2001.

[4] J. F. Canny, “A computational approach to edge detection,” IEEE Transactions on Pattern Analysis and Machine Intelligence vol. 8 pp. 679-698, 1986.

[5] L. D. Cohen, “On Active Contour Models and Balloons,” *Computer Vision, Graphics and Image Processing*: *Image Understanding*, vol. 53, no. 2, pp. 211-218, 1989.

[6] http://www.greencarcongress.com/2010/02/syncrude-to-expand-its-oil-sands-synthetic-crude-output-to-425000-barrels-per-day-by-2020.html. Accessed: June 2011.

[7] L. D. F. Costa and R. M. C. Jr. *Shape Analysis and Classification Theory and Practice*. CRC press, 2001.

[8] Tony F. Chan and Luminita A. Vese, “Active Contours Without Edges,” *IEEE Transactions on Image Processing*, vol. 10, No. 2, pp. 266-277, February, 2001.

[9] Francis H. Y. Chan, F. K. Lam, and Hui Zhu, “Adaptive Thresholding by Variational Method,” *IEEE Transactions on Image Processing*, Vol. 7, No. 3, pp. 468-473, March, 1998.

[10] G. Dong, N. Ray and S. T. Acton, “Intravital leukocyte detection using the gradient inverse coefficient of variation,” *IEEE Tranactions on Medical Imaging*, Vol. 24, pp. 910-924, July 2005.

[11] http://bme.virginia. edu/ley/. Accessed : June, 2011.

[12] D. Freedman and T. Zhang, “Interactive graph cut based segmentation with shape priors,” *CVPR*, pp. 775- 762, 2005.

[13] X. Ge and J. Tian, “An automatic active contour model for multiple objects,” *ICPR*, vol. 2, pp. 881-884, 2002.

[14] R. C. Gonzalez and R. E. Woods, Digitial Image Processing, Second Edition, 2002.

[15] T. Hastie, R. Tibshirani and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer, Second Edition, 2009.

[16] P. Jaccard, **“**Étude comparative de la distribution florale dans une portion des Alpes et Jura,” *Bulletin de la Société Vaudoise des Sciences Naturelles* , vol. 37, pp. 547–579, 1901.

[17] R. Jin, Y. Liu, L. Si, J. Carbonell and A. G. Hauptmann, “A New Boosting Algorithm using Input-Dependent Regularizer,” *International Conference of Machine Learning* (*ICML*), 2003.

[18] I. T. Jolliffe, *Principal Component Analysis*, *Springer*, Second Edition, 2002.

[19] A. E. Johnson and M. Hebert, “Surface matching for object recognition in cluttered 3D scenes,” *IEEE Transactions on PAMI*, vol. 21, no. 5, pp. 433 – 449, 1999.

[20] Z. A. Karian and E. J. Dudewicz, Fitting Statistical Distributions – The Generalized Lambda Distribution and Generalized Bootstrap Methods, CRC Press, 2000.

[21] M. kass, A. Witkins, and Terzopoulos, “Snakes: active contour models,” *IJCV*, vol.1, No.4, pp.321-331, 1987.

[22] B. Li and S. T. Acton, “Automatic active model initialization via Poisson inverse gradient,” *IEEE Transactions on Image Processing*, vol. 17, no. 8, pp. 1406 – 1420, 2008.

[23] B. Li and S. T. Acton, “Active contour external force using vector field convolution for image segmentation,” *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2096 – 2106, 2007.

[24] C. Li, J. Liu and M. D. Fox, “Segmentation of external force field for automatic initialization and splitting of snakes,” *Pattern Recognition*, vol. 38, no. 11, pp. 1947 – 1960, 2005.

[25] Syncrude Canada Limited, 2007 Sustainability Report. Issue: Canadians want responsible development of the oil sand.

[26] C. Liu, “A Bayesian Discriminating Features Method for Face Detection,” *IEEE Transactions on PAMI*, vol. 25, no. 6, pp. 725- 740, June, 2003.

[27] D. Marr and H.K. Nishihara, “Visual information processing: Artificial Intelligence and the sensorium of sight,” *Technology Review*, Vol. 81 no. 1, October 1978.

[28] T. McInerney and D. Terzopoulos, “T-snakes: Topology adaptive snake,” Medical Image Analysis, vol. 4, pp. 73 – 91, 2000.

[29] F. Meyer, “Topographic distance and watershed lines,” *Signal Processing*, vol. 38, pp. 113 – 125, 1994.

[30] M. Mirmehdi, X. Xie and J. Suri. Handbook of Texture Analysis, Imperial College Press, 2008.

[31] Moghaddam and A. Pentland, “Probabilistic Visual Learning for Object Representation,” *IEEE Transactions on PAMI*, vol. 19, no. 17, pp. 696- 710, July, 1997.

[32] J. Morel and S. Solemini, Variational Methods in Image Segmentation, Birkhauser, Boston, 1995.

[33] D. Omerčević, R. Perko, A. T. Targhi, Jan-Olof Eklundh and A. Leonardis, “Vegetation segmentation for boosting performance of MSER feature detector,” *Computer Vision Winter Workshop*, Feb 4-6, 2008.

[34] http://en.wikipedia.org/wiki/Syncrude. Accessed: June 2011.

[35] N. Otsu, “A threshold selection method from gray– level histogram,” *IEEE Transactions on System Man Cybernatics*, Vol. SMC-9, No.1, pp. 62-66, 1979.

[36] N. Ray, S. T. Acton and K. Ley, “Tracking leukocytes in vivo with shape and size constrained active contours,” *IEEE Transactions on Medical Imaging*, vol. 21, no. 10, pp. 1222 - 1235, 2002.

[37] N. Ray and S. T. Acton, “Merging Parametric Active Contours within Homogeneous Image Regions for MRI – Based Lung Segmentation,” *IEEE Transactions on Medical Imaging*, Vol.22, No. 2, 2003.

[38] N. Ray, “Tracking rolling leukocytes in vivo using active contours with motion gradient vector flow,” *PhD Thesis*, University of Virginia, USA, 2003.

[39] N. Ray and S. T. Acton. Biomedical Image Analysis: Tracking. Morgan and Claypool Publishers, 2005.

[40] N. Ray and S. T. Acton. Biomedical Image Analysis: Segmentation. Morgan and Claypool Publishers, 2009.

[41] C. J. van Rijsbergen. Information Retireval. Butterworths, London, 1979.

[42] R. Ronfard, “Region-based strategies for active contour models,” *International Journal of Computer Vision*, vol. 13, no. 2, pp. 229-251, 1994.

[43] P. L. Rosin, “Edges: Saliency measures and automatic thresholding,” *International Geoscience and Remote Sensing Symposium*,” vol. 1, pp. 93 – 95, 1995.

[44] J. C. Russ. The Image Processing Handbook. Third Edition, *CRC & IEEE press*.

[45] B. Saha, N. Ray, “Image thresholding by variational minimax optimization,” *Pattern Recognition*, Volume 42, Issue 5, Pages 843-856, May 2009.

[46] B. N. Saha, N. Ray and H. Zhang, "Computing oil sand particle size distribution by snake-PCA algorithm," *ICASSP*, pp. 977-980, April, 2008.

[47] B. N. Saha, N. Ray, and H. Zhang, "Snake validation: A PCA-based outlier detection method," *IEEE Signal Processing Letters*, vol. 16, issue 6, pp. 549-552, 2009.

[48] B. N. Saha, N. Ray, and H. Zhang, "Automating Snakes for Multiple Objects Detection," Asian *Conference on Computer Vision*, volume 6494/2011, pp. 39-51, 2010.

[49] B. N. Saha, N. Ray, and H. Zhang, "Automated Snake based Segmentation for Multiple Blob-Object Detection," in preparation.

[50] http://scholar.google.ca/scholar?q=active+contour+or+snake&hl=en&as_sdt=1%2C5&as_sdtp=on. Accessed: June, 2011.

[51] V. P. Sharma and J. J. Roger Cheng, Structural health monitoring of Syncrude's Aurora II oil sand crusher, Structural Engineering Report No. 272, University of Alberta Department of Civil & Environmental Engineering, October, 2007.

[52] M. Sonka, Vaclav Hlavac and Roger Boyle, Image Processing, Analysis, and Machine Vision, Third Edition, 2008.

[53] http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/AV0910/SpinImages.pdf. Accessed: June 2011.

[54] Y. Q. Wang, J. Liang and Y. D. Jia, "On the critical point of GVF Snake," ACCV, pp. 754-763, 2007.

[55] Y. T. Xi, Z. J. Xiang, P. J. Ramadge and R. E. Schapire, "Speed and Sparsity of Regularized Boosting," *12th International Conference on Artificial Intelligence and Statistics* (*AISTATS*), April, 2009.

[56] C. Xu and J. L. Prince, "Snakes, Shapes, and Gradient Vector Flow," *IEEE Transactions on Image Processing,* Vol. 7, pp. 359-369, 1998.

[57] C. Xu and J. L. Prince, "Generalized gradient vector flow external forces for active contours," *Signal Processing,* vol. 71, pp. 131 – 139, 1998.

[58] Z. Yu and C. Bajaj, "Normalized Gradient Vector Diffusion and Image Segmentation," *ECCV,* pp. 63-82, 2002.

[59] Hong Zhang, "Image processing for the oil sands mining industry [In the Spotlight]," IEEE Signal Processing Magazine, vol. 25, issue. 6, pp. 198-200, 2008.

[60] X. Zheng and Q. Gao, "Image Noise Removal Using Perceptual Edge Features," *ICGST International Journal on Graphics, Vision and Image Processing,* Special Issue on Edge Detection and Tracking, pp. 1-8, March, 2006.