3D Segmentation for Terrain and Contour

Major Project

Submitted in partial fulfillment of the requirements

For the degree of

Master of Technology in Computer Science and Engineering

By

Vijay Prakash Sharma
(05MCE021)

Guide
Prof. Vibha Patel

Department of Computer Science and Engineering

Institute of Technology

Nirma University of Science and Technology

Ahmedabad 382481

May 2007

This is to certify that Dissertation entitled

3D Segmentation for Terrain and Contour

Submitted by

Vijay Prakash Sharma

has been accepted toward fulfillment of the requirement
for the degree of
Master of Technology in Computer Science & Engineering

Prof. (Dr.) S. N. Pradhan
Professor In Charge

Prof. D. J. Patel
Head of The Department

Prof. A. B. Patel
Director, Institute of Technology

CERTIFICATE

This is to certify that the Major Project entitled “3D Segmentation For Terrain And Contour” submitted by Mr. Vijay Prakash Sharma (05MCE021), towards the partial fulfillment of the requirements for the degree of Master of Technology in Computer Science and Engineering of Nirma University of Science and Technology, Ahmedabad is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project, to the best of my knowledge, haven’t been submitted to any other university or institution for award of any degree or diploma.

Prof. Vibha Patel
Guide,
Professor,
Department of Computer Science and Engineering,
Institute of Technology,
Nirma University,
Ahmedabad.

Date: / /

ACKNOWLEDGEMENT

It gives me great pleasure in expressing thanks and profound gratitude to **Dr. S. N. Pradhan**, M.Tech In-Charge, Department of Computer Engineering, Institute of Technology, Nirma University, Ahmedabad for his valuable guidance and continual encouragement throughout the Major project. I heartily thankful to him for his time to time suggestion and the clarity of the concepts of the topic that helped me a lot during this study.

I would like to give my special thanks to **Prof. Vibha Patel**, Department of Computer Engineering, Institute of Technology, Nirma University for guiding throughout this thesis and providing feedback to improve my work. I would like to give my special thanks to **Prof. D. J. Patel**, Head, Department of Computer Engineering, Institute of Technology, Nirma University for his continual kind words of encouragement and motivation throughout the Major Project. I am also thankful to **Prof A. B. Patel**, Director, Institute of Technology, Nirma University.

I am thankful to all faculty members for their special attention and suggestion towards the project work. I extend my sincere thanks to my colleagues for their support in my work.

I am really thankful to **GOD** who gives me courage to face difficulties and overcome them. I would like to express my gratitude towards my family members who have always been my source of inspiration and motivation.

Vijay Prakash Sharma

(05MCE021)

ABSTRACT

3D segmentation is very important area of image processing. It is used in Medical Imagery, GIS (Geographical Information System) and Satellite Imagery etc.

Segmentation refers to the process of partitioning a digital image into multiple regions. The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries in images.

This thesis work consists of algorithms to segment any type of noisy/blurred image, in this; the approach used for solving this problem is based on active contours. Active contour is a curve that moves towards the sought-for shape. It is controlled by internal and external force of the objects.

It is based on Energy Minimization Algorithms. In this algorithm, based on energy, it recognizes the different objects in image and objects are differentiated on the basis of its boundary.

CONTENTS

LIST OF FIGURE

1. INTRODUCTION

1.1 GENERAL

Vision is the most advanced sense among the five senses of human beings, and plays the most important role in human perception. Although the sensitivity of human vision is limited within the visible band, imaging machines can operate on the images generated by sources that human vision cannot associate with. Thus, machine vision encompasses a wide and varied field of applications, even in areas where human vision cannot function, e.g. infrared (IR), ultraviolet (UV), X-ray, magnetic resonance imaging (MRI), ultrasound.

Although there is no clear distinction among image processing, image analysis, and computer vision, usually they are considered as hierarchies in the processing continuum.

The low-level processing, which involves primitive operations such as noise filtering, contrast enhancement, and image sharpening, is considered as image processing. Note both its inputs and outputs are images.

The mid-level processing, which involves segmentation and pattern classification, is considered as image analysis or image understanding [1]. Note its input generally is images, but its outputs are attributes extracted from those images, e.g. edges, contours, and the identity of individual objects.

The high-level processing, which involves ‘making sense’ of an ensemble of recognized objects and performing the cognitive functions at the far end of the processing continuum, is considered as computer vision [1].

1.1.1 Image Segmentation

In most image analysis operations, pattern classifiers require individual objects to be separated from the image, so the description of those objects can be transformed into a suitable form for computer processing. Image segmentation is

a fundamental task, responsible for the separating operation. The function of segmentation is to partition an image into its constituent and disjoint sub-regions, which are uniform according to their properties, e.g. intensity, color, and texture. Segmentation algorithms are generally based on either discontinuity among sub regions, i.e. edges, or uniformity within a sub-region [1].

The distinction between image segmentation and pattern classification is often not clear. The function of segmentation is simply to partition an image into multiple sub-regions, while the function of pattern classification is to identify the partitioned sub-regions. Thus, segmentation and pattern classification usually functions as separate and sequential processes**.** However, they might function as an integrated process depending on the image analysis problem and the performance of the segmentation method. In either way, segmentation critically affects the results of pattern classification, and often determines the eventual success or failure of the image analysis.

Since segmentation is an important task in image analysis, it is involved in most image analysis applications, particularly those related to pattern classification, e.g. medical imaging, remote sensing, security surveillance, military target detection. The level to which segmentation is carried depends on the problem being solved. That is, segmentation should stop when the region of interest (ROI) in the application have been isolated. Due to this property of problem dependence, autonomous segmentation is one of the most difficult tasks in image analysis. Noise and mixed pixels caused by the poor resolution of sensor images make the segmentation problem even more difficult. In this work, Novel segmentation method using a variational framework, called active contours is presented.

1.1.2 Active Contour

Active contours are connectivity-preserving relaxation methods, applicable to the image segmentation problems. Active contours have been used for image segmentation and boundary tracking. The basic idea is to start with initial boundary shapes represented in a form of closed curves, i.e. contours, and iteratively modify them by applying shrink/expansion operations according to the

constraints of the image. Those shrink/expansion operations, called contour evolution, are performed by the minimization of an energy function [4].

An advantage of active contours as image segmentation methods is that they partition an image into sub-regions with continuous boundaries, while the edge detectors based on threshold or local filtering, e.g. Canny or Sobel operator, often result in discontinuous boundaries.

1.2 MOTIVATION

One interpretation of Active contour is that they represent a top down, rather than bottom-up, approach to image segmentation and edge detection. Common techniques for edge detection are biased toward rectilinear objects. Edge detection typically uses the combination of two separate convolution filters: horizontal and vertical. The bias toward straight edges has been mitigated by other techniques such as the Generalized Hough Transform or template matching. Techniques such as these require an *a* priori description of the object shape to identify the object in imagery. Active contour promises a segmentation technique that does not bias the final shape description toward straight edges. They also perform this segmentation without a known or predetermined shape. Much of the motivation to use active contour in various applications arises from this ability.

Another motivation for the use of active contour is the ability of these models to “fill in” edges where weak image gradients are present. The extreme case for this “filling in” phenomenon would be illusory contours. Consequently, the primary motivation for active contour usage is to capture non-rectilinear (curved) object boundaries and to be robust in the presences of weak edges.

1.3 SCOPE OF WORK

Active contour is a technique for tracing boundaries in an image. It is based on energy minimization concepts. This algorithm can detect objects whose boundaries are not necessarily defined by gradient. Gradient method is prevalent for finding the edges but this method can only detect sharp edges and fails in the

case of smooth edges. It detected the objects and differentiates on the basis of its boundary.

1.4 ORGANIZATION OF MAJOR PROJECT

This thesis is organized as follow:

- Chapter 2 tells about segmentation and basic techniques for segmentation. In this chapter all segmentation techniques are classified into three classes. Structural techniques, stochastic techniques, Hybrid techniques
- Chapter 3 tells basic concepts of active contour and about its energy concepts. Internal energy image energy, concept of minimizes the energy.
- Chapters 4 introduce level set and its segmentation method with edge stopping and with out edge stopping function.
- Chapter 5 is the implementation of without edge stopping function.
- Chapter 6 presents results on different images.

2. SEGMENTATION

2.1 INTRODUCTION

Segmentation subdivides an image in to its constituent regions or objects. The level to which the subdivision is carried depends on the problem being solved. That is, segmentation should stop when the object of interest in an application have been isolated.

Image segmentation algorithms generally are based on one of two basic properties of intensity values:

Discontinuity and similarity

In the first category, the approach is to partition an image based on abrupt changes in intensity, such as edges in an image. The principal approaches in the second category are based on partitioning an image into region that is similar according to a set of predefined criteria. Thresholding, region growing and splitting and merging are example of methods in the category [1].

A large number of algorithms for segmentation can be found in the literature. Due to the nature of the problem of segmentation, most of these algorithms are specific to a particular problem, thus, having little significance for most other problems

All segmentation techniques are classified into three classes.

1. Structural techniques
2. Stochastic techniques
3. Hybrid techniques

2.2 STRUCTURE TECHNIQUES:

Structural techniques try to find structural properties of the region to be segmented. Structural properties such as intersecting surfaces (edges in 2D) are detected in the volume and then combined to segment the region.

2.2.1 3-D Edge-Detection Techniques

It is a type of detection of discontinuities. Edge detection techniques are those which aim at detecting edges or surfaces in the volume to perform segmentation. Edges are formed at the intersection of two regions with different intensities [2].
Edge detection techniques in three dimensions work in two stages:
1. Local edges are detected by using some form of differentiation.
2. These local edges are grouped together to form boundary contours that separate the desired region voxels from other voxels.

One advantage of edge detection techniques is that they work very well on datasets with good contrast between different regions. The edges are detected perfectly and can be verified visually.

These algorithms do not perform well on datasets with low contrast between regions. These algorithms are also susceptible to noise. In most of the cases, these algorithms are not used on their own for segmentation, but coupled with other segmentation algorithms to solve a particular segmentation problem.

Basic formula: An ideal edge has the properties of the model shown in Fig 2.1. An ideal edge according to this model is a set of connected pixels (in the vertical direction here), each of which is located at an orthogonal step transition in gray level (as shown by the horizontal profile in the figure). Edges are more closely modeled as having a "ramp like" profile, such as the one shown in Fig 2.2. The slope of the ramp is inversely proportional to the degree of blurring in the edge [1].

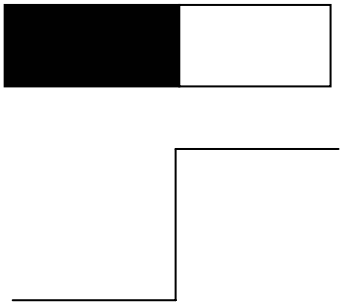

Fig. 2.1Model of an ideal digital edge

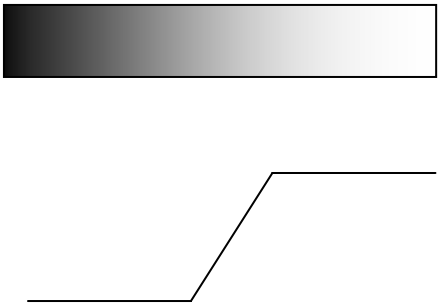

Fig. 2.2.Model of a ramp digital edge

The first derivative is positive at the point of transition into and out of the ramp as move from left to right along the profile; it is constant for the point in the ramp; and is

zero in areas of constant gray level. The second derivative is positive at the transition associated with the dark side of the edge, negative at the transition associated with the light side of the edge, and zero along the ramp and in areas of constant gray level. From these observations, it can be concluded that the magnitude of the first derivative can be used to detect the presence of an edge at a point in an image. Similarly, the sign of the second derivative can be used to determine whether an edge pixel lies on the dark or light side of an edge.

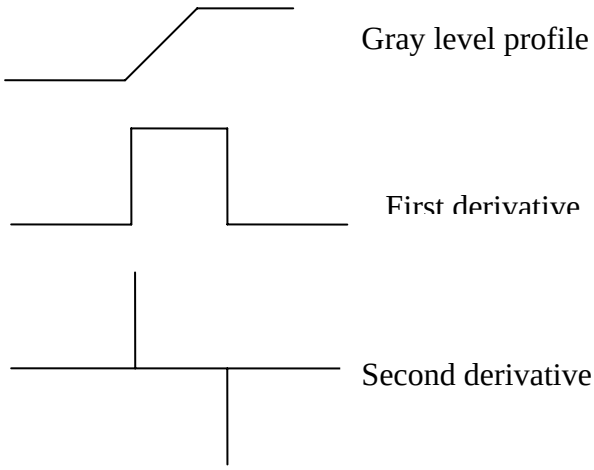

Fig. 2.3 first and second derivatives of gray level profile.

Two additional properties of the second derivative around an edge can be noted:

(1) It produce two values for every edge in an image;

(2) An imaginary straight line joining the extreme positive and negative values of secondary derivative would cross zero near the mid point of the edge.

This zero point crossing property of the second derivative is quite useful for locating the centers of thick edges.

2.2.2 Filters*:*

Gaussian Filter:

For 2D images :

$$Gau[:;:;:] = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

For 3D images:

$$\text{Gau}[:;:;1]=\frac{1}{39}\begin{bmatrix}0 & 1 & 0\\ 1 & 3 & 1\\ 0 & 1 & 0\end{bmatrix}$$

$$\text{Gau}[:;:;2]=\frac{1}{39}\begin{bmatrix}1 & 3 & 1\\ 3 & 9 & 3\\ 1 & 3 & 1\end{bmatrix}$$

$$\text{Gau}[:;:;3]=\frac{1}{39}\begin{bmatrix}0 & 1 & 0\\ 1 & 3 & 1\\ 0 & 1 & 0\end{bmatrix}$$

Sobel filter :

For 2D images:

$$\text{Sob}[:;:;:]=\begin{bmatrix}-1 & 2 & -1\\ 0 & 0 & 0\\ -1 & 2 & -1\end{bmatrix}$$

For 3D images

$$\text{shx}[1;:;:]=\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

$$\text{shx}[2;:;:]=\begin{bmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}$$

$$\text{shx}[3;:;:]=\begin{bmatrix}-1 & -2 & -1\\ -2 & -4 & -2\\ -1 & -2 & -1\end{bmatrix}$$

$$\text{shy}[:;1;:]=\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

$$\text{shy}[:;2;:] = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\text{shy}[:;3;:] = \begin{bmatrix} -1 & -2 & -1 \\ -2 & -4 & -2 \\ -1 & -2 & -1 \end{bmatrix}$$

$$\text{shz}[:;:;1] = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

$$\text{shz}[:;:;2] = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\text{shz}[:;:;3] = \begin{bmatrix} -1 & -2 & -1 \\ -2 & -4 & -2 \\ -1 & -2 & -1 \end{bmatrix}$$

2.2.3 Morphological Techniques

Mathematical morphology uses set transformations for image analysis. It extracts the impact of a particular shape on images via the concept of structuring elements (SE). The SE encodes the primitive shape information. The shape is described as a set of vectors referenced to a particular point, the center [1]. During morphological operations, the center scans the whole image and the matching shape information is used to define the transformation. The transformed image is thus a function of the SE distribution in the whole image. The two most fundamental transforms in mathematical morphology are erosion and dilation. These can be defined on the basis of the above assumptions as

Dilation: With A and B as sets in Z^2, the dilation of A by B, denoted $A \oplus B$

$$A \oplus B = \{ z \mid (B')z \cap A \neq \emptyset \} \qquad \text{----(2.1)}$$

This equation is based on obtaining the reflection of B about its origin and shifting this reflection by z. The dilation of A by B then is the set of all displacements, z, such that B and A overlap by at least one element.

Erosion: For sets A and B in Z^2 the erosion of A by B, denoted A Θ B is defined as

$$A \Theta B = \{z \mid (B)z \subseteq A\} \quad \text{---- (2.2)}$$

In words, this equation indicates that the erosion of A by B is the set of all points z such that, translated by z, is contained in A.

Morphological operations are generally simple to understand and implement. At the same time, these are generally difficult to control. For example, it is difficult to control the dilation operation unless the upper limit to the number of times is given. Thus, these algorithms generally require some external criteria to control them. These operations also have a risk of changing the morphology of the input datasets. It is well known that a series of dilations followed by erode operations leads to loss of high frequencies (for example, folds in a colon), and fills holes. Similarly a series of erodes followed by dilations can introduce holes and high frequencies. These algorithms should be avoided when accuracy is the primary concern and there is a risk of loss of important data. As with edge detectors, morphological operations are not segmentation algorithms by themselves but they are generally an integral part of a segmentation pipeline.

2.2.4 Deformable Models

Deformable models are curves, surfaces or solids defined within an image or volume domain and they deform under the influence of external and internal forces. In the physics-based modeling paradigm, the data apply forces (external forces) to the deformable model and as a result the model moves towards the data, while internal forces keep the model smooth during deformation [3].

Mathematically, a deformable model moves according to its dynamic equations and seeks the minimum of a given energy function. The deformation of a typical 2-D deformable model can be characterized by the following dynamic equation:

$$F_{\text{int}} + F_{ext} = \mu(s)\frac{\partial^2 X(s,t)}{\partial t^2} + \gamma(s)\frac{\partial X(s,t)}{\partial t} \quad \text{-----(2.3)}$$

Where X(s,t) = (x(s, t),y(s, t)) is a parametric representation of the position of the model at a given time t, and μ(s) and $\gamma(s)$ are parameters representing the mass density and damping density of the model, respectively. Equation (2.3) causes the model to move according to the direction and magnitude of the forces on the right hand side.
Internal force represents internal stretching and blending forces. The most commonly used external forces are computed as the gradient of an edge map.

Physically based deformable models can be divided into three categories: energy minimizing snakes, dynamic deformable models, and probabilistic deformable models.

2.2.4.1 Energy minimizing snakes

Snakes [4] are the most popular form of deformable models. Snakes are planar deformable contours that are useful in several image analysis tasks. Using energy minimization formulation, the goal of this approach is to find a parametric model that minimizes the weighted sum of internal energy and potential energy. The internal energy specifies the tension or the smoothness of the surface of the model. The potential energy is defined over the volume domain and typical possesses local minima at the edges occurring at object boundaries. Minimizing the total energy yields internal and potential forces. As a result, these are attracted to image features such as lines and edges.

2.2.4.2 Dynamic deformable models

Although it is natural to think of energy minimization as a static problem, a potent approach to computing the local minima of functional is to construct a dynamical system that is governed by the functional and allow the system to evolve to equilibrium. Equilibrium is achieved when the internal and external forces balance and the contour comes to rest. This leads to dynamic deformable models that unify the description of shape and motion, making it possible to quantify not just static shape, but also shape evolving through time.

2.2.4.3 Deformable models in segmentation

Deformable surface models are also used for segmentation. Initialized a deformable model near the region-of-interest and allowed it to deform into place. Users could then manually fine-tune the fitting by using interactive capabilities of the models. The first uses of deformable models in medical images analysis was the application of deformable contour models, such as snakes, to segment structures in 2D images [6]. To segment 3D medical datasets, each 2D slice was segmented separately. Once a 2D slice was segmented, the contour of that slice was used as a reference contour for neighboring slices. This reference contour was then deformed into place in those slices. This process was repeated for all the 2D slices. The resulting sequence of 2D contours was then connected to form a continuous 3D surface model.

The 3D segmentation process described above is both laborious and requires a post-processing step to connect the sequence of 2D contours into a continuous surface. In addition, the reconstructed surface can have various inconsistencies. A true 3D segmentation technique could overcome all these shortcomings giving smooth 3D surfaces. In one of the initial work on segmentation using 3D deformable surfaces, Miller [7] constructed a balloon by approximating a sphere using polygons. He then geometrically deformed this balloon until its surface conformed to the object surface in 3D image.

The segmentation process is formulated as the minimization of a cost function, which is a weighted sum of three terms: a deformation potential that expands the model vertices towards the object boundary, an image term that identifies features such as edges and opposes the balloon expansion, and a term that maintains the topology of the model by constraining each vertex to remain close to the centroid of its neighbors. Deformable models have the advantage that they offer a coherent and consistent mathematical description and are robust to noise and boundary gaps due to their incorporation of a smoothness constraint. Another advantage is that the offer sub-voxel accuracy for the boundary representation that may be important to a number of applications. A very important advantage of these models from the point of view of medical imaging is that these models are capable of accommodating the often significant variability of biological structures over time and across different individuals.

A disadvantage is that they require manual interaction to place an initial model in the dataset.

2.3 STOCHASTIC TECHNIQUES

The Stochastic techniques algorithms performs segmentation by statistical analysis only, these algorithms do not take into account any structural information.

2.3.1 Thresholding Approaches

Thresholding is probably the simplest of the segmentation techniques [1]. In this technique a single value called threshold is used to create a binary partition of voxel intensities. All voxels with intensities greater than the threshold are grouped together into one class and those with intensities below the threshold are grouped together into another class. Use of a single threshold thus results in a binary segmented volume. This technique can be extended to using multiple thresholds, where a region is defined by two thresholds, a lower threshold and an upper threshold. Each voxel of the input volume then belongs to one of the regions based on its intensity. This technique is known as multi thresholding. In Figure 2.4 we showed histogram of a volume. To apply thresholding, Two thresholds T1 and T2 as shown. We then get three distinct regions as seen from the histogram.

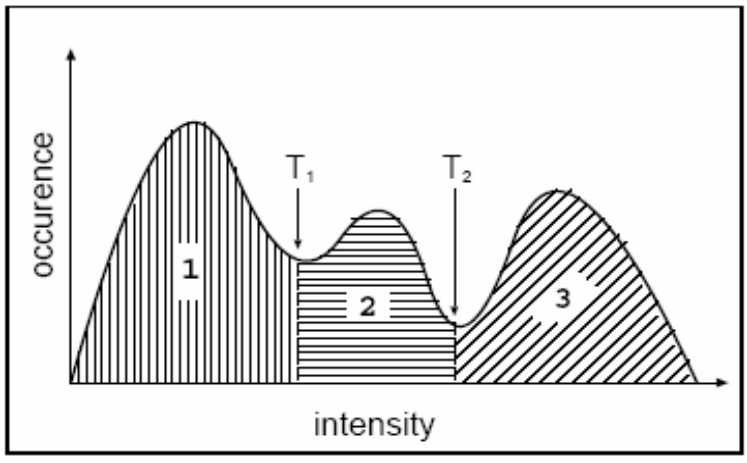

Fig. 2.4 Histogram of a volume with two thresholds T1 and T2 dividing the histogram in three regions.

Although simple, this technique is very effective in getting segmentation done in volume with a very good contrast between regions. This is generally used as the first step towards segmentation of a volume. The main drawback of this technique is that

the results are too tightly coupled with the thresholds used. Any change in the threshold values can give a different segmented region. The thresholds are usually generated interactively by using visual feedback. Some automatic methods do exist with varying degree of success to automate the process of finding correct thresholds. Another drawback which is a direct consequence of the previous one is that the technique is very sensitive to noise and intensity in homogeneities.

2.3.2 Classification Techniques

Classification techniques are pattern recognition techniques [8] that seek to partition a feature space derived from the volume using data with known labels. A feature space is the range of an N-dimensional feature vector made from features at each voxel. The features could include the voxel intensity, the gradient at the voxel, the distance of the voxel from the volume boundary and so on. Mathematically, a feature space can be the range space of any function of the volume. Classifiers belong to the supervised category as they require training data that are pre segmented (either manually or by other method). The pre-segmented data is then used as reference to carry out automatic segmentation on new data. The simplest form of a classifier is the nearest neighbor classifier, where each pixel or voxel is classified in the same class as the training datum with the closest intensity. The k-nearest neighbor (kNN) classifier is the generalization of this approach, where the pixel is classified according to the majority of the k closest training data. Another example of a similar classifier is the Parzen window, where the classification is made according to the majority vote within a predefined window of the feature space centered at the unlabeled voxel (mapped to feature space). Both these classifiers are non-parametric since they don't make any assumption about the statistical structure of the data.

Disadvantage of classifier technique is the manual interaction for obtaining training data. Training sets can be acquired from each volume that requires segmentation, but this can be time consuming and laborious. On the other hand, use of the same training set for a large number of scans can lead to biased results which do not take into account anatomical and physiological variability between different subjects.

2.3.3 Clustering Algorithms

These are clustering-based techniques [9] which use characteristics of the voxel and its immediate neighborhood to do clustering. Clustering can be loosely defined as the process of grouping objects into groups, whose members show similar properties. In our case these "objects" are the data voxels and the "groups" are the segmented regions. "Similar properties" could be any property the data voxel posses, like the density, gradient, color (for a color dataset) etc. Clustering-based segmentation is similar to the classifier methods, with the exception that these do not use any training data. These techniques thus come under the unsupervised class of algorithms for segmentation. These algorithms overcome the need for a training data by iterating between segmenting the volume and characterizing the properties of each class. We could say that clustering-based algorithms train themselves using the available data.
The various clustering algorithms available today can be grouped into two broad categories:

1. Hierarchical methods: These methods include those techniques where the input data is not partitioned into clusters in a single step. A series of successive fusions of data are performed until each cluster of size greater than one is composed of smaller clusters.

2. Non-Hierarchical methods: In these methods, the desired number of clusters is known or assumed at the beginning of the clustering process. The end result is such that each data voxel gets assigned to exactly one cluster in this algorithm.

As in the case of classification, voxel properties such as intensity, gradient, neighborhood information etc. are used to form an N-dimensional feature vector for each voxel. Each class of the region is assumed to form a distinct cluster in the N-dimensional feature space. A suitable clustering algorithm, (K-means clustering, leader clustering, spatial clustering, etc.) is then applied to each voxel in the feature space. The resultant clusters in the feature space are then mapped to spatial domains to give the desired regions [9].

2.3.3.1 K-means clustering

This algorithm takes as input a set of N dimensional vectors without any prior knowledge about the set. After processing, the algorithm forms K disjoint nonempty subsets such that each subset minimizes some measure of dissimilarity. By minimizing dissimilarity of each subset locally, the algorithm will globally yield an optimal dissimilarity of all subsets. The dissimilarity for a voxel is its distance from the mean of each of the classes in the feature space. The mean for each class is computed iteratively. The voxel is added to the cluster whose mean is the nearest to the voxel (meaning least dissimilarity between the voxel and the cluster's mean).
The algorithm has a time complexity O(RKN), where K is the number of desired clusters, and R is the number of iterations until it converges.

2.3.3.2 Fuzzy clustering

The input to the algorithm is a finite data set $X = x_1; x_2;...;x_n$, each $x_i \epsilon X$ is a feature vector; $x_i = (x_{i1}; x_{i2};....;x_{is})$ where x_{ij} is the j^{th} feature of subset x_i, and s is the dimensionality of x_i. A function $u: X \rightarrow [0; 1]$ is defined, which assigns to each x_i in X its grade of membership in the fuzzy set u. The function u is called a fuzzy subset of X. The goal is to partition X by means of fuzzy sets. A fuzzy c-partition is defined as c X n matrix U such that:

1. Each row U_i represents the i^{th} fuzzy subset of X.
2. Each column U^j exhibits the membership grades of datum j in every fuzzy subset.
3. The membership grades of each datum in all fuzzy subsets add up to 1.
4. No fuzzy subset is empty.
5. No fuzzy subset is all of X.

2.4 HYBRID APPROACHES

Segmentation algorithms, which cannot be classified into the previous two categories, are discussed here. In this section segmentation techniques based on finding the region directly [1].
Basic formulation: let R represent the entire image region. R is partitions into n sub regions, R_1, R_2 , R_3,……, R_n. such that

a) $\bigcup_{i=1}^{n} Ri = R$

b) Ri is a connected region, i=1,2,3…n.

c) $Ri \cap Rj =$ Ø for all I and j, i≠j.

d) $P(Ri) = TRUE$ for i=1,2…n.

e) $P(Ri \cup Rj) = FALSE$ for i≠j

Here, $P(R_i)$ is a logical predicate defined over the points in set R_i and Ø is the null set. Condition (a) indicates that the segmentation must be complete; that is, every pixel must be in a region. Condition (b) requires that points in a region must be connected in some predefined sense. Condition (c) indicates that the region must be disjoint. Condition (d) deals with the properties that must be satisfied by the pixels in segmented region. Condition (e) indicated that regions R_i and R_j are different in the sense of predicate P.

2.4.1 Region Growing

This is probably the simplest among the hybrid techniques. Region growing is a technique to extract a connected region from a 3D volume based on some pre-defined connecting criterion. These criteria can be as simple as the voxel intensity. In the simplest form, region growing requires a seed point to start with. From the seed point, the algorithm grows till the connecting criteria are satisfied. As with thresholding, region growing is simple, but not often used for segmentation by itself. More often than not, region growing forms a part of a segmentation pipeline for a particular approach. It is often used as the primary. The primary disadvantage of this algorithm is that it requires seed points which generally mean manual interaction. Thus for each region to be segmented, a seed point is needed. Region growing can also be sensitive to noise and partial volume effect causing the extracted region to have holes or disconnections. Some recent work has been reported which tries to alleviate these problems.

2.4.2 Split and Merge

This algorithm is similar to region growing algorithm. This algorithm requires the input data to be organized into a pyramidal grid structure of regions, with each region

organized in group of eight. Any region can be split into eight sub regions and the appropriate eight regions can be merged into a single larger region. As in region growing, the criteria for merging (growing for region-growing) could be anything. It could be as simple as voxel intensity or some condition checking based on the output of some previous segmentation stage. Let us assume that the criterion is P.
The algorithm can be written down in two steps as follows:
1. Pick a region R in the grid structure. If P(R) is false, split the region into sub regions. If for eight regions R1;R2; :::;R8, C(R1UR2U…..UR8) = TRUE, merge into single region. When no regions can be merged, stop.
2. If there are neighboring regions Ri and Rj such that $P(R_i \cup R_j)$=TRUE, merge these regions.

The big advantage of this method over region growing is that no seed points are needed and hence no manual interaction is needed.

2.4.3 Artificial Neural Networks

Conventional segmentation algorithms based on structural knowledge often require considerable user expertise. The Artificial neural networks (ANN) [10] based approaches tried to partially overcome these drawbacks. ANNs are massively parallel networks of processing elements or nodes that simulate biological learning. Each node in an ANN is capable of performing elementary computation. Learning is achieved through the adaptation of weights assigned to the connections between nodes.
The main features of ANNs which the segmentation algorithms try to use are:
1. Learning from examples and generalizing that knowledge
2. Noise rejection
3. Fault tolerance
4. Optimum seeking behavior

Three architectures for image segmentation based on ANNs are available. These architectures showed that ANNs can successfully exploit and integrate different kinds of a priori information contained in medical images. His experiments demonstrated robustness and sensitivity of the approach, but at the expense of generality. ANNs are widely used in segmentation as a classifier, where the weights are determined using training data, and the ANN is then used to segment new data. ANNs can also be used

in an unsupervised fashion as a clustering method, as well as for deformable models. Since the ANNs are tightly interconnected, spatial information can be easily incorporated into its classification procedures. Although ANNs are inherently parallel, their processing is usually simulated on a standard serial computer, thus reducing this potential computational advantage.

3. ACTIVE CONTOUR

3.1 INTRODUCTION

In the late eighties it was suggested that it should be possible to follow edges in images by suggesting a curve (e.g., the circumference of an object) in an image, and then letting the curve itself move to a suitable shape and position. This curve should have physical properties like elasticity and rigidity, and also be attracted by edges in the image. Such curves are called active contours, deformable models or snakes and have become popular especially in medical image analysis [12].

For the contour to be attracted to edges in the image an energy image or attractor image is created, which has high values where the original image has edges and low values otherwise. The attractor image gives the contour a potential energy by summing the energy in the points the contour passes. The contour itself has an internal energy level determined by its shape (elasticity and rigidity) and by minimizing the total energy one aims at a smooth contour that follows the original image's edges well [11].

Fitting active contours to shapes in images is an interactive process. The operator must suggest an initial contour, which is quite close to the intended shape. The contour will then be attracted to features in the image extracted by creating an attractor image.

3.1.1 Open and closed contours

The contour can be either a closed or an open curve. If the contour is open, one should take care to modify the contour's definition of its energy so that the end-points will not move in the same way as the other points (avoiding the contour dragging itself into itself and vanishing).

3.2 THE ENERGY OF THE CONTOUR

The energy depends on the shape of the contour (internal energy) and on its positioning on the image according to

$$E(v,f) = E_{image}(v,f) + E_{\text{int}}(v) \qquad \text{------ (3.1)}$$

These energies influence all points along the contour with internal forces and an image force. When all forces are balanced, the total energy is minimum.

3.2.1 Internal Energy

The internal energy of the contour depends on the shape of the contour and the parameter functions α(s) and β(s) and is defined as

$$E_{\text{int}} = \int \left(\alpha(s).\left|v'(s)\right|^2 + \beta(s).\left|v''(s)\right|^2\right)ds \qquad \text{------- (3.2)}$$

The first term, $|v'(s)|^2$ will have larger values if there is a large gap between successive points on the contour and minimizing it will minimize the total length of the contour. The second term, $|v''(s)|^2$, will be larger where the contour is bending and requires the contour to be as smooth as possible. This term is weighted by parameter functions, and so α(s) determines the elasticity of the contour, and β(s) determines the rigidity [12, 13].

If α(s) equals zero at some points then discontinuities are allowed there, and where β(s) equals zero, discontinuous curvature such as corners are allowed.

3.2.2 Image energy

The image energy depends on how the contour is positioned on an attractor image *p*, and it is defined as

$$E_{image} = -\int p(v(s),f)ds \qquad \text{------ (3.3)}$$

Where $$p(x,f) = |\nabla f(x)|^2$$ ------ (3.4)

3.2.3 Minimize the energy

Minimizing the energy Equation (3.1) is equivalent to solving the corresponding Euler-Lagrange-equation

$$\alpha.v'' - \beta.v^{(4)} = -\nabla p(v,f)$$ ----- (3.5)

This simply means that the internal forces shall balance the image force.
In practice one does not study the contour at all points. Instead, the contour is represented by a vector $\bar{v}$ of control points v_j. The control points must not be separated by more than a few pixels to prevent the contour from bypassing attractive but small areas in the image. It is the purpose of the elasticity force to keep the control points equidistant. Control points v_j the derivatives in Equation (3.5) can be approximated as

$$v_j'' \approx v_{j-1} - 2v_j + v_{j+1}$$ ----- (3.6)

$$v_j^{(4)} \approx v_{j-2} - 4v_{j-1} + 6v_j - 4v_{j+1} + v_{j+2}$$ -----(3.7)

Using these expressions Equation (3.5) can be written as

$$A\bar{v} + \nabla p(\bar{v},f) = 0$$ -----(3.8)

Where A is the matrix given by Equation (3.6) and Equation (3.7) and insertion of α and β. A contour can then be calculated with iterations according to Euler's method as

$$(I + A)\bar{v}_{k+1} = \bar{v}_k + \nabla p(\bar{v}_k,f)$$ ------ (3.9)

3.3 THE CONTOUR-FITTING PROCESS

To find a suitably shaped and positioned contour three main steps are executed [11]

Step 1: Suggesting an initial contour: To make the contour attract to the shape in the image to which one wants to fit the contour, an initial suggestion should be given. A 2D contour is simply represented by a vector of control points where neighboring control points are represented by neighboring elements in the vector. When segmenting a 2D image, the representation is straightforward; a vector of control points.

Step 2: Calculating the attractor image: The original image has to be filtered to create the attractor image according to Equation (3.4). The attractor image is then filtered with gradient filters to create the image force ∇p. The image is filtered with two edge detecting filters (detecting edges in different directions) and in two scales. The results are added, normalized, and squared,

Step 3: Iterate: The suggested contour is iterated according to Equation (3.9). The iteration will proceed until it eventually gives a stable contour or until the user interrupts the iteration.

3.4 THE ATTRACTOR IMAGE

As mentioned earlier the original attractor image was given by

$$p(x,f) = |\nabla f(x)|^2 \qquad \text{---- (3.10)}$$

And the image force on the contour *v* at a point s ε [0 1] is given by $\nabla p(v(s), f)$.
This attractor image has some limitations:

- It is only sensitive to edges, not to lines. That implies that the contour will be attracted to points on either side of a line. This is not always a problem, but in some applications it certainly is.
- In 3D, it is sensitive to lines as well as to planes. When trying to segment a volume in a 3D image.

The first problem is quite easily solved by taking the magnitude of an orientation estimate on the original image.

To solve the second problem it is suitable to create a local estimate of the probability of linear and planar structures.

3.4.1 Attracting to local orientation in 2d images

By filtering an image with edge and line detectors in four directions, it is possible to estimate the orientation vector as follows:
If the convolution kernel $q_{line}(x)$ is a horizontal line detecting filter kernel and $q_{edge}(x)$ is a horizontal edge detecting filter kernel, then

$$q_1(x,f) = |(f * q_{line})(x) + i(f * q_{edge})(x)| \quad \text{----- (3.11)}$$

is an estimate of the dominance of local horizontal orientation in the image *f*. Combining *q1* with a "vertical estimate" *q3*, one can get a more secure estimate in q1-q3, since vertical and horizontal orientation are regarded as opposites. This is an estimate of the local one-dimensionality of the neighborhood, i.e., that the neighborhood contains structure oriented horizontally only (a negative value means that the neighborhood is dominated by vertical structure). Using q_1, q_3 and their rotations by Π/4 q_2 and q_4 the orientation vector can be created as

$$z = \begin{bmatrix} q_1 - q_3 \\ q_2 - q_4 \end{bmatrix} \quad \text{------ (3.12)}$$

An estimate of the local magnitude of orientation can be calculated as the norm of z, which equals

$$|z|^2 = \sum_{k=1}^{4} q_k^2 - 2(q_1 q_3 + q_2 q_4) \quad \text{----- (3.13)}$$

Using this norm

$$p(x,f) = |z(x,f)|^2 \quad \text{-----(3.14)}$$

as the attractor image will cause the contour to be attracted to oriented structure invariantly to phase, i.e., to lines as well as to edges. it is not really necessary with four complex filters; in two dimensions it is quite enough using three filters

detecting lines/edges in directions separated by Π/3. The norm can then be calculated as

$$|z|^2 = \sum_{k=1}^{3} q_k^2 - q_1 q_2 + q_1 q_3 + q_2 q_3 \qquad \text{-----(3.15)}$$

3.4.2 Attracting to planar structure in 3d images

Let Orientation tensor ***T***, whose eigen system describes the local variation of the signal (or the local energy distribution in the Fourier domain) and therefore also the orientation. Calling ***T***'s eigenvalues λ1, λ2, λ3 where λ1, λ2, λ3 and the corresponding eigenvectors *e1*, *e2* and *e3*, there are (in 3D) three characteristic. cases:

• Plane case: λ1>> λ2 ≅ λ3 ≅ *0*

The signal varies mainly in one direction, *e1*, i.e the spatial neighborhood describes a planar structure, with *e1* as normal vector (the plane is spanned by *e2* and *e3*). The energy in the Fourier domain is mainly distributed along a line in the direction of *e1*.

• Line case: λ 1≅ λ 2 >> λ 3≅ *0*

The energy in the Fourier domain is distributed on a plane spanned by *e1* and *e2*. The spatial neighborhood is (nearly) constant in the same direction and describes a linear structure along *e3*.

• Isotropic case: λ1 ≅ λ 2≅ λ 3

The neighborhood has no orientation, i.e it varies equally in all directions.

Using a certainty estimate as attractor

The spectrum theorem states that orientation

$$T = \lambda_1 . e_1 . e_1^T + \lambda_2 . e_2 . e_2^T + \lambda_3 . e_3 . e_3^T \qquad \text{----- (3.16)}$$

And since $e_2 e_2^T \perp e_3 e_3^T$ the norm $\Delta(T)$, can be calculated as

$$\Delta(T) = \left\| T - T_1 \right\| = \lambda_2 . e_2 . e_2^T + \lambda_3 . e_3 . e_3^T \qquad \text{----- (3.17)}$$

$\Delta(T)$ will be close to zero in the presence of planar structure, and computation of Equation. in each point in the image creates an attractor image that will attract the contour to planar structures in the image.

Δ (T) will also be close to zero when all eigen values are small. Consequently is very sensitive to noise on low-energy parts of the image, and therefore a division by λ_1 is suitable.
The attractor image can then be expressed as

$$P_{\Delta}(x,f) = -\Delta_{\lambda}(\text{eigen values } (T(x,f))) \quad \text{----- (3.18)}$$

Where

$$\Delta_{\lambda}(\lambda) = \frac{\sqrt{\lambda_2 + \lambda_3}}{\lambda_1}$$

3.4.3 Creating the attractor image

As indicated above, several steps are required to create the attractor image and the resulting image force. The sequence of these steps is described below (a 3D image is assumed).
1. Tensor filtering: Filter the image with at least six edge/line detectors and create the orientation tensor.

2. Averaging: The tensor field usually needs some averaging (low-pass-filtering) to reduce randomness of the orientation caused by noise. An ordinary or Gaussian kernel is used.

3. Computing eigenvalues: Compute the eigenvalues of the tensor field and sort them in decreasing order.

4. Detecting planar structure: Compute the energy according to Equation (3.18).

5. Thresholding: The parts of the original image without dominant local orientation, e.g black background, will (especially in the presence of noise) contribute to the calculated attractor, however with quite low values. It is,

though, preferable to set all these points to zero. This will straighten out the contour where crossing such regions, since the contour will there only be under the influence of the internal forces (elasticity and rigidity).

6. Gradient filtering: Computing the 3D-gradient of the attractor image will result in the image force used when iterating the contour. The image force is a function, i.e., every "pixel" in the 3D-image is a three-element vector.

3.5 REPRESENTATION IN THREE DIMENSIONS

3.5.1 The 3D Contour

A 3D contour is typically described by a function $v:[0,1]X[0,1]\rightarrow R^3$. The contour is placed on an image f: $R^3 \rightarrow R$ and is moved by iteration in the same way as in the 2D case. The simplest way to create an active contour in 3D is by using repeated 2D contours. This method is often used since it is also quite intuitive in many applications with image sequences. The major drawback is that it is impossible to vary the distance between the control points in the third dimension, so that exactly one point per pixel (in the third dimension) is required. In some applications one does not even care about elasticity and rigidity in the third dimension and calculates a number of independent contours on separate images. This, of course, can result in very unpleasant contours, e.g., if a piece of a line through the image sequence is missing in one or some images.

3.5.2 The Energy Of The Contour

In 3D the image force is defined in the same way as in 2D, but the internal energy has to be calculated in a slightly different way, To calculate the internal energy in 3D, some additional parameter functions are required, and the energy is now defined as (with v = v(s, r))

$$E_{\text{int}} = \int (\alpha_s \left|v'_s\right|^2 + \alpha_r \left|v'_r\right|^2 + \beta_s \left|v''_{ss}\right|^2 + \beta_r \left|v''_{rr}\right|^2 + \beta_{sr} \left|v''_{sr}\right|^2) dsdr \quad \text{----- (3.19)}$$

Where α_s and α_r determine the elasticity along the s- and r-axis respectively, β_s and β_r determine the corresponding rigidities and β_{sr} determines the resistance to twist.

3.5.3 Representation

A 2D contour is simply represented by a vector of control points where neighboring control points are represented by neighboring elements in the vector. In 3D, the contour is most easily represented by a matrix of control points.

Representing the contour with a polyhedron: To create a sphere-like contour, problem is control points will then be placed on the sphere in a quite irregular manner (with regard to their mutual distances), and there is no obvious way to modify eq.1 to fit this topology. Consequently the mesh-representation is not suitable for spherical contours - instead it is preferable to place the control points regularly on a sphere, i.e., to let the control points be the vertices of a regular polyhedron (in case of eight control points, they should be placed like the corners of a cube). A solution is to create a quasi-regular polyhedron with equal distances between each vertex and its neighbors. Such polyhedron can be created recursively from an icosahedron, see Figure 3.1, using the following algorithm:

1. Place a new vertex at the midpoint of each edge.
2. Remove the old edges and create new edges between all vertices with a mutual distance that equals half of the distance between the old vertices.
3. Project the new vertices on the circum sphere of the polyhedron.

The original twelve vertices have five neighbors while all new vertices will have six neighbors. The number of faces, edges and vertices (F_n, E_n and V_n where n is the number of recursions) grow quite quickly. The original polyhedron P_0 has got the characteristics (F_0 E_0 V_0) = (20 30 12), and the characteristics of polyhedra P_n is calculated recursively (directly from the algorithm) as

$$\begin{aligned} F_n &= 4F_{n-1} \\ E_n &= 2E_{n-1} + 3E_{n-1} \\ V_n &= V_{n-1} + E_{n-1} \end{aligned} \qquad \text{------ (3.20)}$$

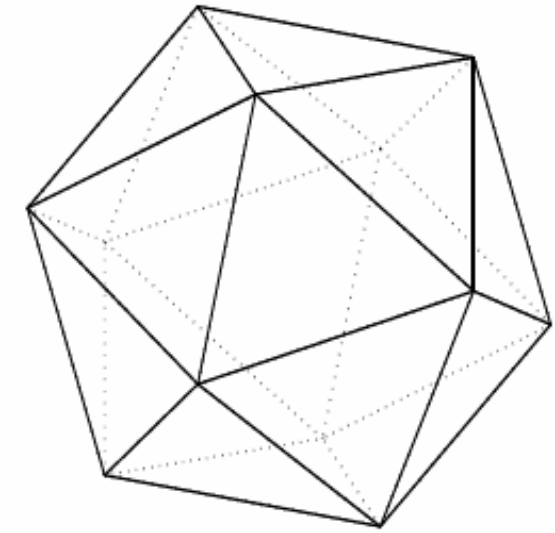

Figure 3.1 The icosahedron, one of the five Platonic polyhedra.

Expanding En as

$$E_n = 2*(2*E_{n-2} + 3*F_{n-2}) + 3*F_{n-1}$$
$$E_n = 2*(2*(E_{n-3} + 3*F_{n-3}) + 3*F_{n-2}) + 3*F_{n-1} \quad \text{------ (3.21)}$$

Makes it obvious that En can be written as a closed expression:

$$E_n = 2^n * E_0 + 3 * \sum_{k=0}^{n-1} 2^{n-1-k} * F_k$$
$$= 2^n * E_0 + 3 * 2^{n-1} * \sum 2^{-k} F_0 * 4^k \quad \text{------ (3.22)}$$
$$= 30 * 4^k$$

In a similar way

$$F_n = 20 * 4^n$$
$$E_n = 30 * 4^n \quad \text{----- (3.23)}$$
$$V_n = 2 + 10 * 4^n$$

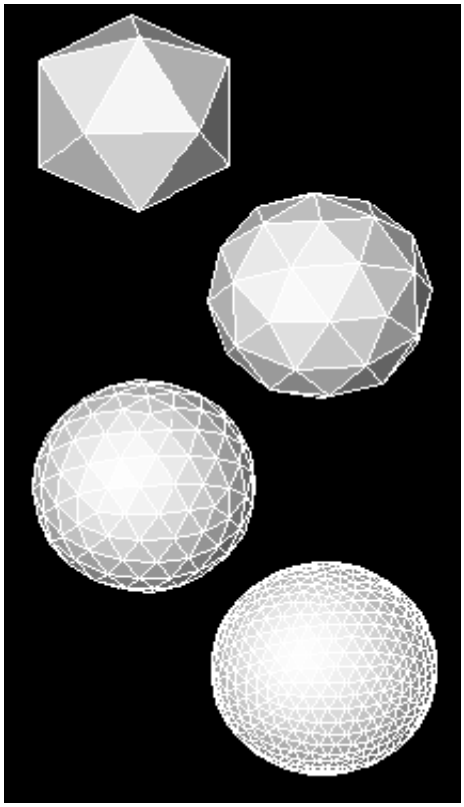

Figure 3.2 The icosahedron and its first three expansions.

For example, P3 and P4 have 642 and 2562 vertices respectively, which could be suitable amounts in practical applications. The four first polyhedra (P0 through P3) are illustrated in Figure 3.2.

3.5.4 The Internal Forces Independent of Representation

Elasticity can be regarded as a dragging force from each of the neighboring control points, i.e., the elasticity force in a control point is simply the vector from the control point to the average of the neighboring control points (Figure 3.3).

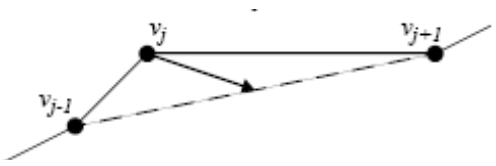

Figure 3.3 The elasticity force of the control points.

Rigidity can be regarded as the force from a control point to a point linearly "predicted" by the two "earlier" points, as shown in Figure 3.4, i.e., the vector

from v_{j-2} to v_{j-1} is added to v_{j-1}, and v_j is forced towards the resulting point. This means that the rigidity force in a point v_j is . The "prediction" can be made in all directions and be generalized to higher dimensionality.

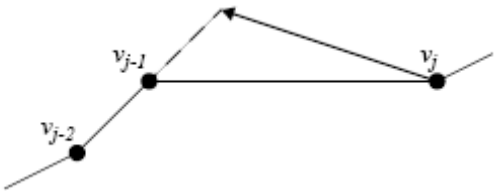

Figure 3.4 The rigidity force of the control points.

4. ACTIVE CONTOUR USING LEVEL SETS FOR SEGMENTATION

4.1 LEVEL SET METHOD

The level set method is a numerical technique for tracking interfaces and shapes. The advantage of the level set method is that one can perform numerical computations involving curves and surface on a fixed Cartesian grid without having to parameterize these objects (this is called the Eulerian approach). Also, the level set method makes it very easy to follow shapes which change topology, for example when a shape splits in two, develops holes, or the reverse of these operations [13]

4.1.1 Mathematical Formulation

Let Ω be a bounded open subset of R^2, with ∂Ω as its boundary. Then a two dimensional image u0 can be defined as u0: Ω→ R. In this case Ω is just a fixed rectangular grid. Now consider the evolving curve C in Ω, as the boundary of an open subset ω of Ω. In other words, ω of Ω and C is the boundary of ω (C =∂ω).

The main idea is to embed this propagating curve as the zero level set of a higher dimensional function Φ. A function is defined as follows:

$$\varphi(x,y,t=0) = \pm d \qquad \text{---- (4.1)}$$

Where d is the distance from (x, y) to ∂ω at t = 0, and the plus (minus) sign is chosen if the point (x; y) is outside (inside) the subset ω [14].

Now, the goal is to produce an equation for the evolution of the curve. Evolving the curve in the direction of its normal amounts to solving the partial differential equation:

$$\frac{\partial \Phi}{\partial t} = F\left|\nabla \Phi\right|,\ \Phi(x,y,0) = \Phi_0(x,y) \qquad \text{---- (4.2)}$$

Where the set {(x, y), Φ_0(x, y) =0} defines the initial contour, and F is the speed of propagation. For certain forms of the speed function F, this reduces to a standard Hamilton-Jacobi equation. There are several major advantages to this formulation. The first is that Φ(x, y, t) always remains a function as long as F is smooth. As the surface Φ evolves, the curve C may break, merge, and change topology.

Another advantage is that geometric properties of the curve are easily determined from a particular level set of the surface Φ. For example, the normal vector for any point on the curve C is given by:

$$\bar{n} = \nabla \Phi$$

And the curvature K is obtained from the divergence of the gradient of the unit normal vector to the front:

$$K = div\left(\frac{\nabla \Phi}{|\nabla \Phi|}\right) = \frac{\Phi_{xx}\Phi_y^2 - 2\Phi_x\Phi_y\Phi_{xy} + \Phi_{yy}\Phi_x^2}{(\Phi_x^2 + \Phi_y^2)^{\frac{3}{2}}} \qquad \text{----- (4.3)}$$

Finally, another advantage is that this system is able to evolve curves in dimensions higher than two. The above formulae can be easily extended to deal with higher dimensions. This is useful in propagating a curve to segment volume data.

4.2 ACTIVE CONTOURS WITH EDGE STOPPING

The goal now is to define a speed function F from the data.

F = FA + FG.

FA represents a constant advection term that will force the curve to expand or contract uniformly based on its sign. This acts like the inflation force used in traditional snakes models. The second term FG depends on the geometry of the curve and acts to smooth out high curvature regions [14].

However, A method is required to halt the evolution of the curve at object boundaries. Let assume that boundaries are defined by the gradient of image u_0, then

$$g(u_0) = \frac{1}{1 + \left|\nabla G_\sigma(x,y) * u_0(x,y)\right|^p}, p \geq 1 \qquad \text{------- (4.4)}$$

Where $G_\sigma(x,y) * u_0(x,y)$ is simply the convolution of u0 with the Gaussian function

$$G_\sigma = \sigma^{-1/2} e^{-\left|x^2+y^2\right|/4\sigma}$$

The function $g(u_0)$ has values that are close to zero in regions where the gradient of the image is high, and values that are closer to one in homogeneous regions. Another edge stopping function which falls to zero faster about the edges can be defined as:

$$g(u_0) = e^{-\left|\nabla G_\sigma(x,y) * u_0(x,y)\right|} \qquad \text{------ (4.5)}$$

Multiply speed function F by this edge stopping function g. So Equation (4.2) can be rewritten to reflect the above changes:

$$\frac{\partial \Phi}{\partial t} = g(u_0)\left(F_A\left|\nabla \Phi\right| + F_G\left|\nabla \Phi\right|\right), \Phi(x,y,0) = \Phi_0(x,y) \qquad \text{------ (4.6)}$$

Now for discretize Φ, differences scheme is used. Let Δt be the time step and (x_i, y_j) be the grid points for $1 \leq i; j \leq M$. Let $\Phi^n_{i,j} = \Phi(x_i, y_j, n\delta t)$ be an approximation of Φ(x, y, t) with $n \geq 0$, $\Phi^0 = \Phi_0$. Finite differences are described by the following notation:

$$\begin{aligned} \Delta^x_- \Phi_{i,j} &= \Phi_{i,j} - \Phi_{i-1,j} \quad & \Delta^x_+ \Phi_{i,j} &= \Phi_{i+1,j} - \Phi_{i,j} \\ \Delta^y_- \Phi_{i,j} &= \Phi_{i,j} - \Phi_{i,j-1} \quad & \Delta^y_+ \Phi_{i,j} &= \Phi_{i,j+1} - \Phi_{i,j} \end{aligned} \qquad \text{------ (4.7)}$$

Take F_A=1; use upwind schemes first introduced by Sethian [9] to estimate $F_A \mid \nabla\Phi \mid$:

$$F_A\left|\nabla\Phi\right| = \left[\max\left(\Delta_-^x \Phi_{i,j}^n, 0\right)^2 + \min\left(\Delta_+^x \Phi_{i,j}^n, 0\right)^2 + \max\left(\Delta_-^y \Phi_{i,j}^n, 0\right)^2 + \min\left(\Delta_+^y \Phi_{i,j}^n, 0\right)^2\right]^{\frac{1}{2}} \quad \text{---- (4.8)}$$

The remainder force F_G is based on the K. Multiply this by a small negative constant -ϵ and approximate $\left|\nabla\Phi\right|$ using central differences. So an estimate for $F_G\left|\nabla\Phi\right|$ is:

$$F_G\left|\nabla\Phi\right| = -\varepsilon K\left[\left(\frac{\Phi_{i+1,j}^n - \Phi_{i-1,j}^n}{2}\right)^2 + \left(\frac{\Phi_{i,j+1}^n - \Phi_{i,j-1}^n}{2}\right)^2\right]^{\frac{1}{2}} \quad \text{---- (4.9)}$$

Although the level set equation of motion for Φ is defined over the entire domain, the image-based speed term has meaning only on the curve C. To remedy this, a speed function F′ required, which is globally defined. Let Q be a point on {Φ =c} where c≠ 0 (i.e. Q is not on the curve C), and let P be the closest point to Q such that P is on {Φ =0} (i.e. P is on the curve C). Then the speed at Q should take on the value of the speed at P. The edge-stopping function should be similarly extended to g′. From Equation (4.6)

$$\Phi_{i,j}^{n+1} = \Phi_{i,j}^n - \Delta t[g'(u_0)(F_A'\left|\nabla\Phi\right| + F_G'\left|\nabla\Phi\right|)] \quad \text{----- (4.10)}$$

ALGORITHM:

Initialize Φ^0 by Φ_0, n=0

For fixed number of iterations do

For each (x_i, y_j)ϵ C do

Compute speed $F'_{i,j}$ by (4.8) and (4.9)

Multiply speed by $F'_{i,j}$ stopping function $g'(u_0)$

End

For each (x_i,y_j) ∉ C do

Find point (x′,y′) ϵ C closest to (x_i,y_j)

Extend speed and edge stopping function

End

Compute Φ^{n+1} by (4.10)

End

4.3 ACTIVE CONTOURS WITHOUT EDGES

Assume that image u_0 is formed by two regions of approximately constant intensities u_0^1 and u_0^0, and the object to be detected is represented by the region with value u_0^i. If the boundary is given by C_0, then $u_0 \approx u_0^i$ inside C_0 and $u_0 \approx u_0^o$ outside C_0. The following fitting energy

$$F_1(c) + F_2(c) = \int_{inside\ c} |u_0 - c_1|^2 dx + \int_{outside\ c} |u_0 - c_2|^2 dx \quad \text{----- (4.11)}$$

Where C is any variable curve, c_1 and c_2 are average of u_0 inside and outside of C respectively. Consider Fig. 4.1. If the curve C is outside the object, then $F_1(c)>0$, $F_2(c)\approx 0$. If the curve is inside the object, then $F_1(c) \approx 0$, $F_2(c)>0$. If the curve is both inside and outside the object, then $F_1(c)>0$, $F_2(c) >0$. However, if the curve C is exactly on our object boundary C0, then $F_1(c) \approx 0$, $F_2(c) \approx 0$ and our fitting term is minimized [15].

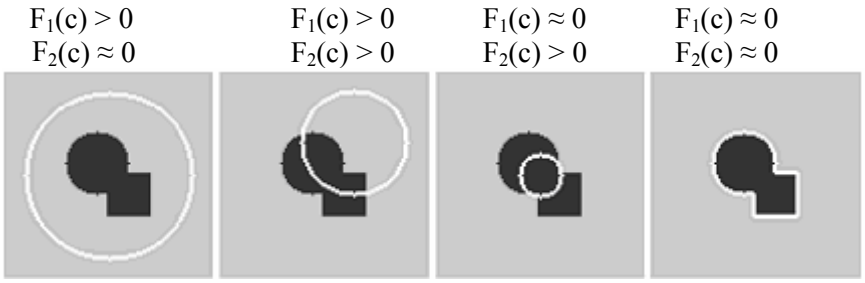

Figure4.1 All possible cases in position of the curve.

Adding some regularizing terms like the length of C and the area inside C, the energy function F(C; c1; c2) is given by

$$\begin{aligned} F(C,c_1,c_2) &= \mu(length\ of\ C)^p + \nu(area\ inside\ C) \\ &+ \lambda_1 \int_{inside\ C} |u_0 - c_1|^2 dx + \lambda_2 \int_{outside\ C} |u_0 - c_2|^2 dx \end{aligned} \quad \text{----- (4.12)}$$

Where $\mu \geq 0, \nu \geq 0, \lambda_1, \lambda_2 > 0$ are fixed parameters. So our goal is to find C,c_1,c_2 such that F (C,c_1,c_2) is minimized.

This problem can be formulated using level sets as follows. In the level set method, C is represented by the zero level set of a Lipschitz function $\Phi : R^N \rightarrow R$ such that

$$\begin{aligned} C &= \{x \in R^N : \varphi(x) = 0\} \\ inside\ C &= \{x \in R^N : \Phi(x) > 0\} \\ outside\ C &= \{x \in R^N : \Phi(x) < 0\} \end{aligned} \qquad \text{------ (4.13)}$$

Now consider the Heaviside function H, and the Dirac measure δ

$$H(z) = \begin{cases} 1 & if\ z \geq 0 \\ 0 & if\ z \leq 0 \end{cases} \text{ and } \delta(z) = \frac{d}{dz} H(z)$$

The Heaviside function is positive inside our curve and zero elsewhere, so the area of the region is just the integral of the Heaviside function of ϕ. The gradient of the Heaviside function defines our curve, so integrating over this region gives the length of the curve.

Using the standard definition for the Heaviside function H and the dirac measure δ,

$$length\ of\ C = \int_{\Omega} |\nabla H(\Phi)| dx = \int \delta(\Phi) |\nabla \Phi| dx \qquad \text{------ (4.14)}$$

Area inside $C = \int_{\Omega} H(\phi) dx$ thus

$$\int_{inside\ C} |u_0 - c_1|^2 dx = \int_{\Omega} |u_0 - c_1|^2 H(\Phi) dx \qquad \text{----- (4.15)}$$

$$\int_{outside\ C} |u_0 - c_2|^2 dx = \int_{\Omega} |u_0 - c_1|^2 \{1\text{-}H(\Phi)\} dx \qquad \text{----- (4.16)}$$

Therefore

$$\begin{aligned} F(\Phi, c_1, c_2) = {} & \mu \left(\int_{\Omega} \delta(\Phi) |\nabla \Phi| dx \right)^p + \nu \int_{\Omega} H(\Phi) dx \\ & + \lambda_1 \int_{\Omega} |u_0 - c_1|^2 H(\Phi) dx + \lambda_2 \int_{\Omega} |u_0 - c_1|^2 \{1 - H(\Phi)\} dx \end{aligned} \qquad \text{----- (4.17)}$$

Minimizing the energy functional with respect to c1 and c2 gives

$$c_1(\Phi) = \frac{\int_\Omega u_0 H(\Phi) dx}{\int_\Omega H(\Phi) dx} \quad \text{------ (4.18)}$$

$$c_2(\Phi) = \frac{\int_\Omega u_0 \{1 - H(\Phi)\} dx}{\int_\Omega \{1 - H(\Phi)\} dx} \quad \text{------- (4.19)}$$

Which correspond to the average value of u0 inside C and outside C respectively, Now deduce the Euler-Lagrange partial differential equation from Equation (4.17).

$$\frac{\partial \Phi}{\partial t} = \delta(\Phi)[\mu\, div(\frac{\nabla \Phi}{|\nabla \Phi|}) - v - \lambda_1(u_0 - c_1)^2 + \lambda_2(\mu_0 - c_2)^2] = 0 \text{ ---- (4.20)}$$

For solving this partial differential equation, regularized H(z) and $\delta(s)$ required

$$H_e(z) = \frac{1}{2}\left(1 + \frac{2}{\Pi}\arctan(\frac{z}{\varepsilon})\right)$$

$$\delta_e(z) = H_e'(z) = \frac{1}{\Pi}(\frac{\varepsilon}{\varepsilon^2 + z^2})$$

with the help of Equation (4.3), Equation (4.19) can be written as

$$\frac{\Phi^{n+1} - \Phi^n}{\Delta t} = \delta_e(\Phi^n)[\mu K - v - \lambda_1(u_0 - c_1)^2 + \lambda_2(u_0 - c_2)^2] \quad \text{------ (4.21)}$$

Algorithm:

Initialize Φ^0 by Φ_0, n=0

For fixed number of iteration do

Compute c_1 and c_2 by Equation (4.18) and (4.19)

Compute curvature term K by Equation (4.3)

Compute Φ^{n+1} by Equation (4.21)

End

5. IMPLEMENTATION OF ACTIVE CONTOUR WITHOUT EDGE STOPPING

5.1 INPUT:

1. Select the image for segmentation.
2. Choose a center point in image.
3. Create a circle surrounding the interesting object.

Fig 5.1 Creation the circle

5.2 FUNCTIONS

5.2.1 Initphi

Pass this image matrix to Initphi function. This function calculates the distance of each pixel from center point. And divide the whole image into two parts. One is outside the circle and second one inside the circle. It set positive sign for pixels outside the boundary and negative for pixels inside the boundary. It returns a matrix called “phi”.

5.2.2 Isfront

Input of Isfront function is “phi” matrix, output of Initphi function. This function returns a binary matrix whose value at each pixel represent whether the corresponding pixel in the phi is a front point or not. The function evaluate

consecutive value of matrix “phi” continually, and if there is any difference between sign then it represent as a front point.

5.2.3 Createimage

Image matrix and phi matrix are inputs of createimage function. It draws the segmented region in green overtop of image. The region is determined by finding the front points from phi.

5.2.4 Calcenergy

This function calculates the energy of inside and outside region of the circle. This function also calls heaviside function and dirac delta function.

$$Inside\ \text{energy} = \left(\mathrm{u}_0 - c_1\right)^2 \qquad \text{------ (5.1)}$$

$$outside\ \text{energy} = \left(\mathrm{u}_0 - c_2\right)^2 \qquad \text{------- (5.2)}$$

Where

$$C_1 = sum(sum(u_0 * H\_phi))/sum(sum(H\_phi)) \qquad \text{------ (5.3)}$$

$$C_2 = sum(sum(u_0 * H\_phi\_minus))/sum(sum(H\_phi\_minus)) \qquad \text{----- (5.4)}$$

H_phi is heaviside value of distance matrix and

H_phi_minus = 1- H_phi

After calculating inside and outside energy calculate total energy

$$energy\_term = nu - lambda1 * \text{inside} + \text{lambda}\,2 * \text{outside} \qquad \text{------ (5.5)}$$

Where lambda 1 =1
Where lambda 2 =1

5.2.4 Heaviside step function

This function takes the value of distance matrix and converts to range from 0 to 1. So all values lies between 0 and 1. Distance matrix is passed in this function [A.2].

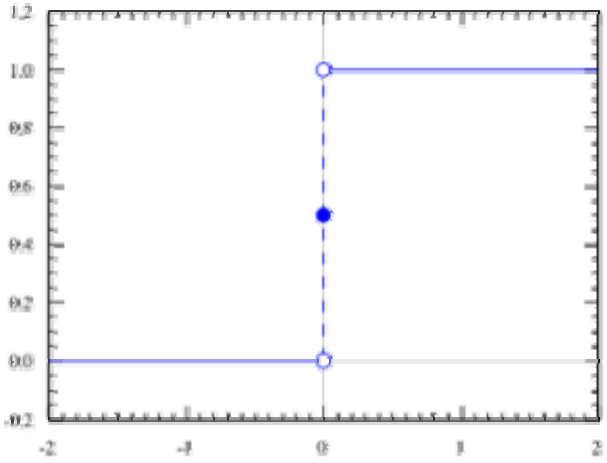

Fig 5.2 PDF of heaviside function

Representation of Heaviside step function

$$value = 0.5 * (1 + 1(2/\Pi) * a\tan(z/var)) \quad \text{----- (5.6)}$$

$$value = \exp(-\exp(-z/var)) \quad \text{----- (5.7)}$$

$$value = 1/(1 + \exp(-z/var)) \quad \text{----- (5.8)}$$

Where var =1, and z is distance matrix

If Equation (5.6) is chosen for heaviside function:

Figure 5.3 Output by using equation (5.6) (a) Creation of circle (b) After one iteration

If Equation (5.7) is chosen:

Figure 5.4 Output by using equation (5.7) (a) Creation of circle (b) After one iteration

If Equation (5.8) is chosen:

Figure 5.5 Output by using equation (5.8) (a) Creation of circle (b) After one iteration

So all three function are available for heaviside function. And for this project Equation (5.6) is chosen.

5.2.5 Dirac delta function

Dirac function is use to Compute the derivative of the heaviside.

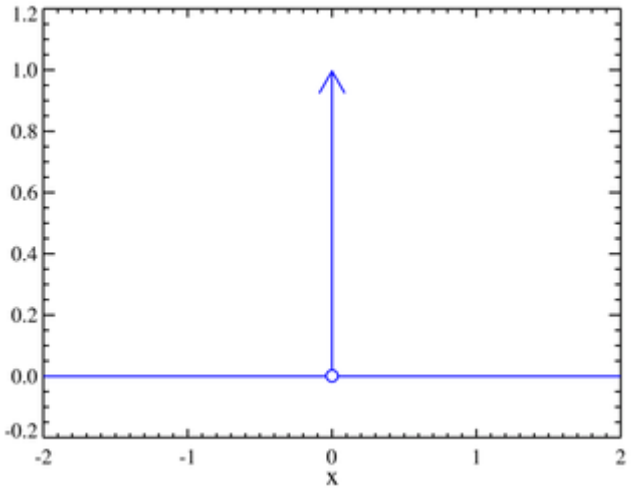

Fig 5.6 PDF of Dirac Delta function

Representation of Dirac Delta function [A.3]:

$$y = (1/\Pi) * (var/(var^2 + x^2)) \quad \text{------ (5.9)}$$

$$y = \left[1/(var * \sqrt{\Pi})\right] * \left[\exp(-x^2)/var^2\right] \quad \text{----- (5.10)}$$

$$y = (1/\Pi * x) * \sin(x/var) \quad \text{----- (5.11)}$$

where var = 1

If Equation (5.9) is chosen for Dirac delta function:

Figure 5.7 Output by using equation (5.9) (a) Creation of circle (b) After one iteration

If Equation (5.10) is chosen for Dirac delta function:

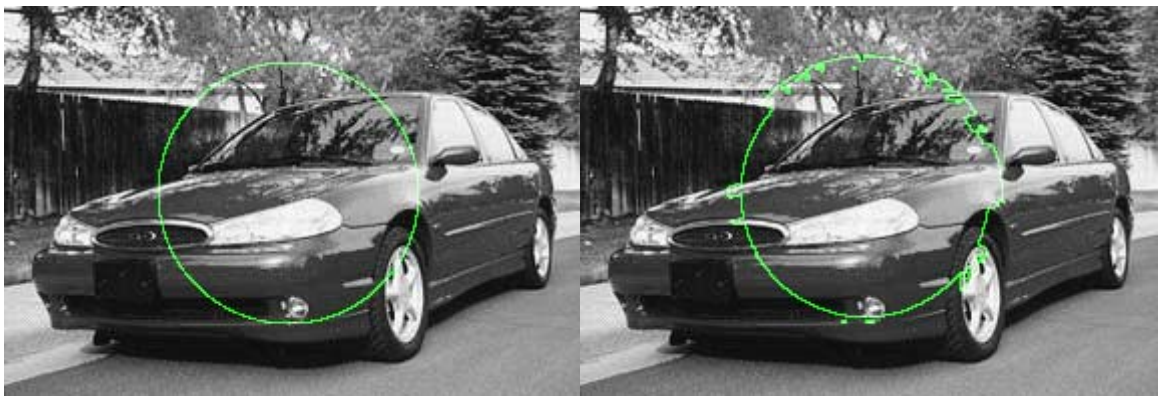

Figure 5.8 Output by using equation (5.10) (a) Creation of circle (b) After one iteration

If Equation (5.11) is chosen for Dirac delta function: It gives an error. Because at some point x=0 (dividing by zero).

That's why Equation (5.9) is chose for it because with Equation (5.10) changes are very less and thus it will take lot of time to execute, and in Equation (5.11) it gives an error.

6.1 RESULT 1

6.1.1 step 1.Select the image

Fig 6.1 Input Image 1

6.1.2 Step 2.Select center point and create the circle

Fig 6.2 Circle Created For Image 1

6.1.3 Result after 50 iteration

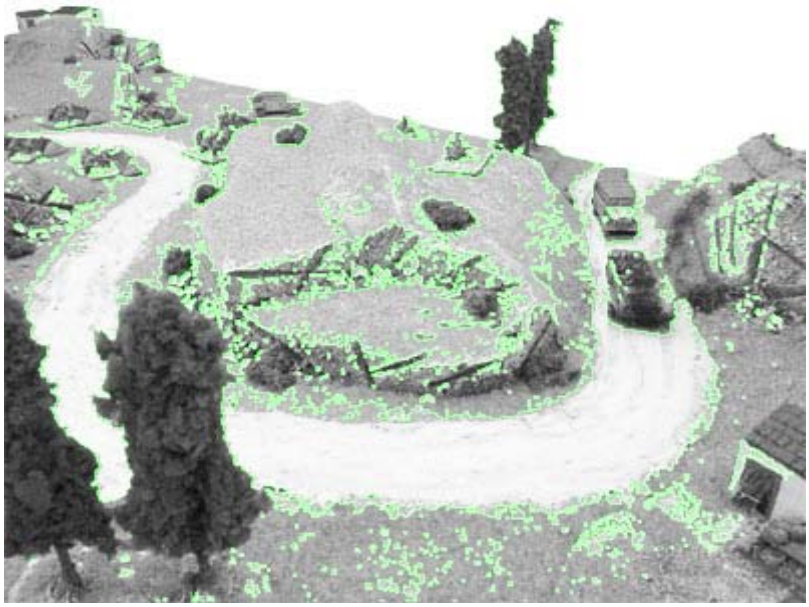

Fig 6.3 Output after 50 Iterations for Image 1

6.1.4 After 110 iteration

Fig 6.4 Fully segmented Image 1

6.2 RESULT 2

6.2.1 step 1. Select the image

Fig 6.5 Input Image 2

6.2.2 Step 2. Select center and create circle

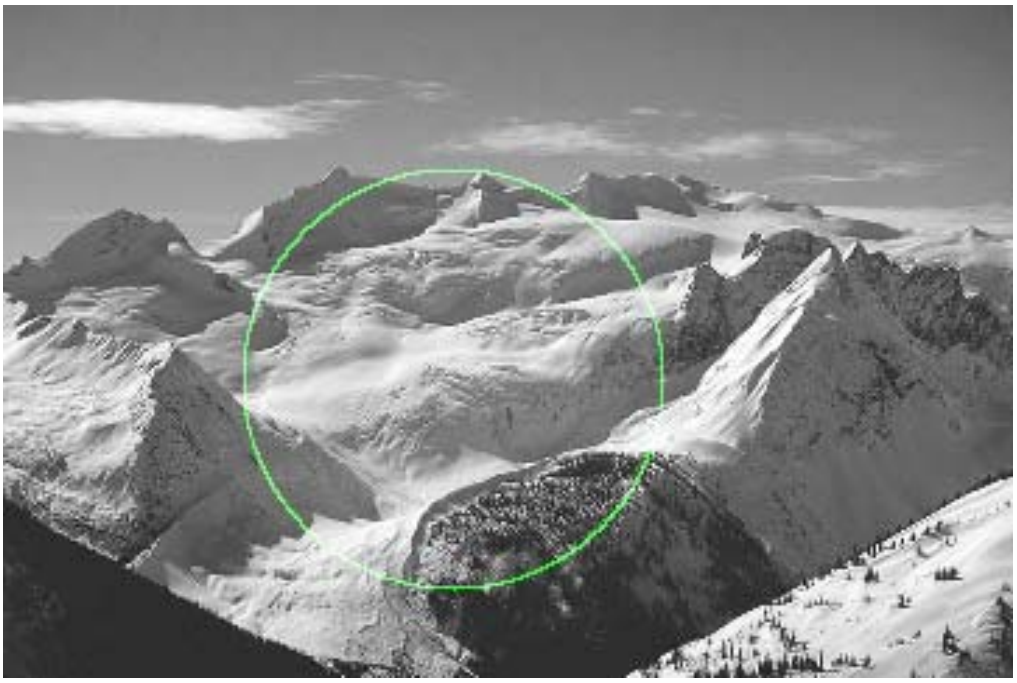

Fig 6.6 Circle Created For Image 2

6.2.3 Result after 70 iteration

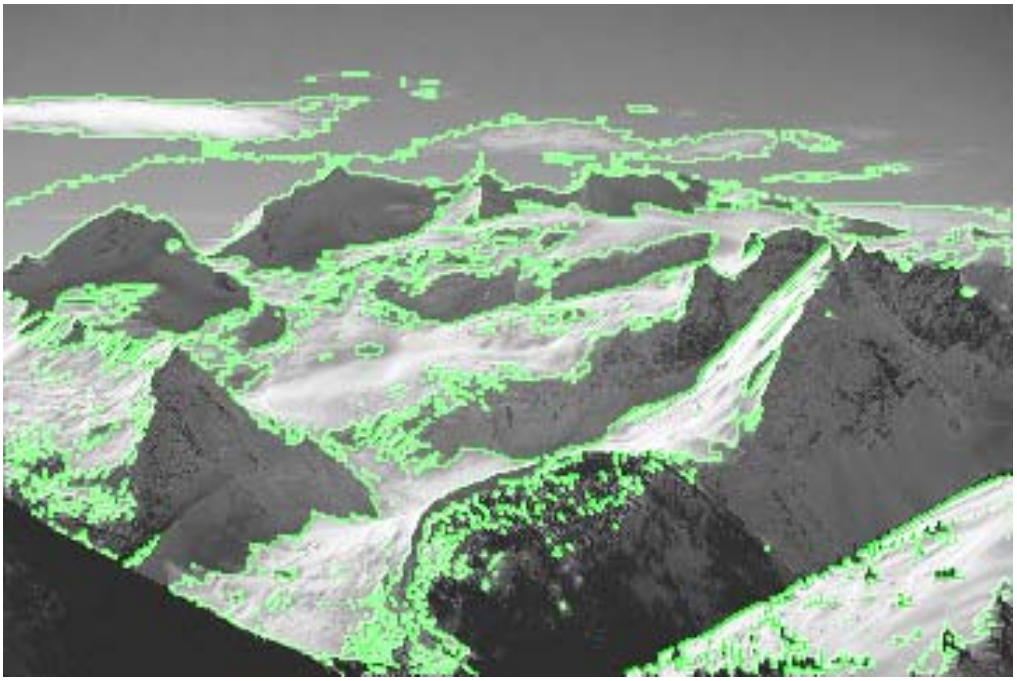

fig 6.7 Output after 70 Iterations for Image 2

6.2.4 Segmented image

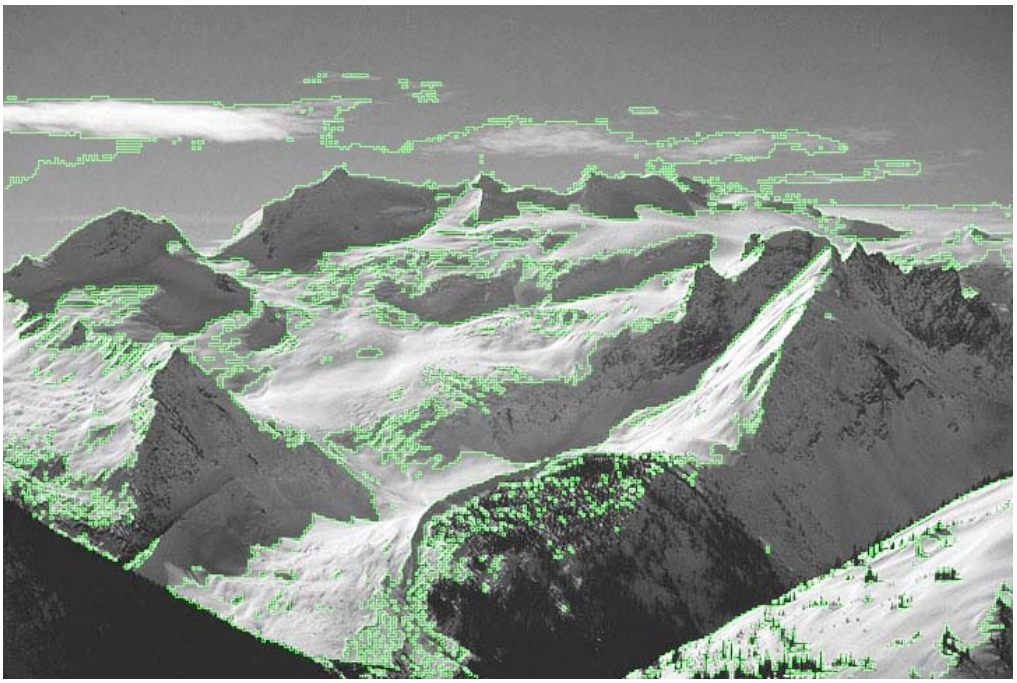

Fig 6.8 Fully segmented Image 2

6.3 RESULT 3

6.3.1 Step 1. Select the image

Fig 6.9 Input Image 2

6.3.2 step 2. Select center and create circle

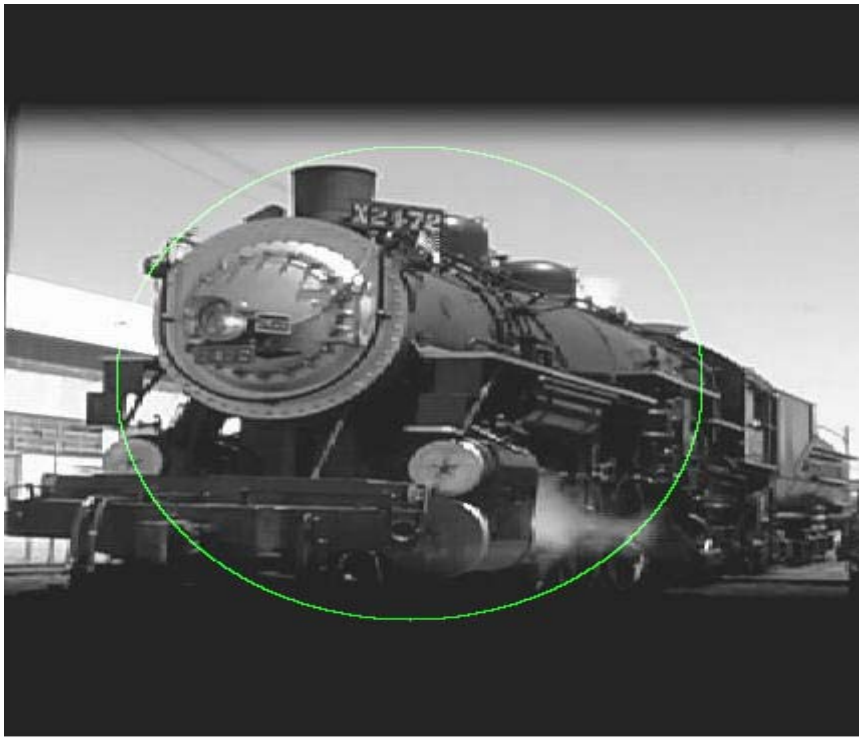

Fig 6.10 Circle Created For Image 3

6.3.3 Result after 52 iteration

Fig 6.11 Output after 52 Iterations for Image 3

6.3.4 Segmented image

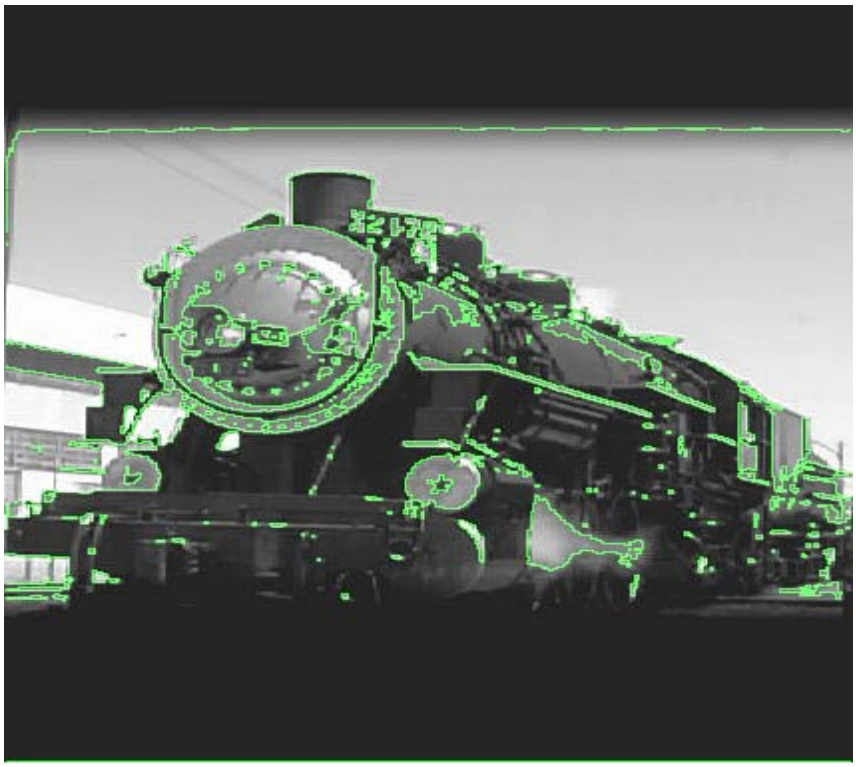

Fig 6.12 Fully segmented Image 3

7. CONCLUSION

Segmentation is an important part of image processing. It is used for dividing the image in constituent regions or objects. The system proposed uses active contour for the segmentation. It is used to trace the boundaries of objects in image. It is a curve that move toward the sought for shape. The system is based on energy minimization. When internal energy and image force becomes equal then energy will be minimum. It can also detect the smooth edges, which are not possible with gradient. This algorithm will terminate when difference between two consecutive images will less then a fixed threshold value. This threshold value depends on number of pixels on edges.

REFERENCES

1. R. Gonzales and R. Woods, Digital Image Processing. Prentice Hall, 2nd ed., 2001.

2. H. K. Liu. Two and Three Dimensional Boundary Detection. Computer Graphics and Image Processing, 6:123{134, Apr. 1977.

3. D. Terzopoulos and K. Fleischer. Deformable Models. The Visual Computer, 4(6):306{331, Dec. 1988}.

4. M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. International Journal of Conputer Vision, 1(4):321{331, 1988}.

5. R. Szeliski. Bayesian modeling of uncertainty in low-level vision. International Journal on Computer Vision, 5:271{301, 1990}.

6. L. D. Cohen. On active contour models and balloons. CVGIP: Image Understanding, 53(2):211{218, 1991}.

7. J. V. Miller, D. E. Breen, W. E. Lorensen, R. M. O'Bara, and M. J. Wozny. Geometrically Deformed Models: A Method for Extracting Closed Geometric Models from Volume Data. Computer Graphics (Proc. SIGGRAPH '91 Conf.), 25(4):217{226, 1991}.

8. Classification techniques in pattern recognition, Lihong heng and XiangiianHe.

9. G. B. Coleman and H. C. Andrews. Image segmentation by clustering. In Proc. IEEE, pages 773{785, 1979}

10. Artificial Neural Network based Segmentation and apply Grading By Machine Vision, Devrim unay Bernard Gosselin.

11. Active Contours in Three Dimensions, by Jorgen Ahlberg, Reg.no. LiTh-ISY - 1708

12. Active contour and their utilization at image segmentation, Mariam Bakos.

13. The level set method, Carolyn L Philips, MIT undergraduate journal of Mathematics.

14. Active contour using level sets for Medical image segmentation, by Michael Wasilewski.

15. Level set framework for image segmentation, by Benjamin ong, simon Fraser University.

APPENDIX A LIST OF USEFUL WEBSITES

A.1. Active contour model (snakes):
http://www.cs.sfu.ca/~stella/papers/blairthesis/main/node28.html

A.2. Heaviside Step function
mathworld.wolfram.com/HeavisideStepFunction.html

A.3 Delta Function:
mathworld.wolfram.com/DeltaFunction.html