2004

Morphological Image Segmentation for Co-Aligned Multiple Images Using Watersheds Transformation

Hyun Geun Yu

THE FLORIDA STATE UNIVERSITY

COLLEGE OF ENGINEERING

MORPHOLOGICAL IMAGE SEGMENTATION FOR CO-ALIGNED MULTIPLE IMAGES USING WATERSHEDS TRANSFORMATION

By

HYUN GEUN YU

A thesis submitted to the
Department of Electrical and Computer Engineering
in partial fulfillment of the
requirements for the degree of
Master of Science

Degree Awarded:
Fall Semester, 2004

The members of the Committee approve the thesis of Hyun Geun Yu defended on November 5, 2004.

_____
Rodney G. Roberts
Professor Directing Thesis

_____
Simon Y. Foo
Committee Member

_____
Anke Meyer-Baese
Committee Member

Approved:

_____
Leonard J. Tung, Chair, Electrical and Computer Engineering

_____
Ching-Jen Chen, Dean, College of Engineering

The Office of Graduate Studies has verified and approved the above named committee members.

This is dedicated to my wife and son, Youmi and Isaac.
Thank you for your existence.

ACKNOWLEDGEMENTS

First of all, I would like to thank Dr. Rodney G. Roberts for his assistance in this research and his constant support throughout my graduate career, and Dr. Simon Y. Foo and Dr. Anke Meyer-Baese for there gracious service on my graduate committee. I would also like to thank Dr. Emmanuel Collins and all CTA project members for their cooperation. I am very grateful to my best colleague, Edmond Dupont, for all the assistance that he has given me both professionally and nonprofessionally.

Finally, I would like to thank my parents and my wife for their support, faith and love.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Image segmentation is one of the most important categories of image processing. The purpose of image segmentation is to divide an original image into homogeneous regions. It can be applied as a pre-processing stage for other image processing methods. There exist several approaches for image segmentation methods for image processing. The watersheds transformation is studied in this thesis as a particular method of a region-based approach to the segmentation of an image. The complete transformation incorporates a pre-processing and post-processing stage that deals with embedded problems such as edge ambiguity and the output of a large number of regions. Multiscale Morphological Gradient (MMG) and Region Adjacency Graph (RAG) are two methods that are pre-processing and post-processing stages, respectively. RAG incorporates dissimilarity criteria to merge adjacent homogeneous regions.

In this thesis, the proposed system has been applied to a set of co-aligned images, which include a pair of intensity and range images. It is expected that the hidden edges within the intensity image can be detected by observing range data or vice versa. Also it is expected that the contribution of the range image in region merging can compensate for the dominance of shadows within the intensity image regardless of the original intensity of the object.

CHAPTER 1

INTRODUCTION

Image processing and analysis is an important area in the field of robotics. This is particularly true for the operation of autonomous vehicles. The operation of an autonomous vehicle is based on first acquiring data that describe its environment. Indeed, the motion planning and control of a fully autonomous vehicle requires an intelligent controller to be able to make decisions to allow the autonomous vehicle to maneuver in an unknown field based on these data. These data sets include range data, 2D images, and position measurements. This data is used to identify and avoid obstacles and to map the surrounding terrain.

The elements of an image analysis system are shown in Figure 1. Image analysis usually starts with a pre-processing stage, which includes operations such as noise reduction. For the actual recognition stage, *segmentation* should be done before it to extract out only the part that has useful information. Image segmentation is a primary and critical component of image analysis. The quality of the final results of an image analysis could depend on the segmentation step. On the other hand, segmentation is one of the most difficult tasks in image processing, especially automatic image segmentation.

The goal of the segmentation process is to define areas within the image that have some properties that make them homogeneous. The definition of those properties should satisfy the general condition that the union of neighboring regions should not be homogeneous if we consider the same set of properties. After segmentation, we can usually establish that the discontinuities in the image correspond to boundaries between regions.

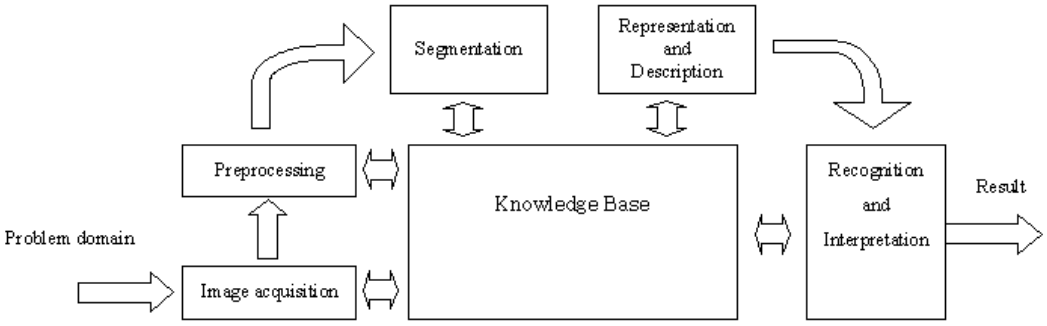


Figure 1.1: Elements of Image Analysis

The methods most commonly used for image segmentation can be categorized into four classes.

1. ***Edge-based approaches****:* Image edges are detected and then linked into contours that represent the boundaries of image objects. The main advantage of edge-based approaches is their lower computational cost. However, the edge grouping process presents serious difficulties in setting appropriate thresholds and producing connected, one-pixel-wide contours.

2. ***Clustering-based approaches****:* Image pixels are sorted in increasing order as a histogram according to their intensity values. Fuzzy-c-means (FCM) and K-means fall into this method. The main advantage of this approach is that the problem of setting thresholds can be avoided by using iterative processes. Also, the segmented contours are always continuous. But, oversegmentation may occur because pixels in the same cluster may not be adjacent.

3. ***Region-based approaches****:* The goal is the detection of regions that satisfy a certain predefined homogeneity threshold. Region-based approaches are available because the segmented contours are always continuous and one-pixel-wide. The computation time of this approach is short. However, different similarity threshold

settings may lead to different segmentation results. Also it can cause oversegmentation.

4. ***Split/merge approaches****:* An input image is first segmented into homogeneous primitive regions using K-means or FCM as a 'Split' step. Then, similar neighboring regions are merged according to a certain decision rule as a 'Merge' step.

In this thesis, we examine a segmentation procedure based on morphological techniques, which is one method of region-based approaches. Mathematical morphology is a branch of nonlinear signal processing and is a powerful tool for the geometrical shape description and analysis of images. In Chapter 2, basic morphological operations are explained with mathematical concepts and examples. The watersheds transformation is described in Chapter 3. Its history and mathematical definitions are presented with some illustrations. Some of shortcomings of the watersheds transformation are also discussed. Chapters 4 and 5 are dedicated to the improvements of the watersheds transformation as 'pre' and 'post' processing. The morphological gradient is reviewed as a noise-insensitive method to calculate the gradient of an image and is compared to the conventional gradient method. The morphological gradient is extended to a multiscale for greater robustness in Chapter 4. The region merging algorithm using region adjacency graph (RAG) is introduced in Chapter 5. Also, some of the differences are mentioned between Split/Merge image segmentation and the region merging using RAG. Chapter 6 consists of a possible image segmentation application motivated from autonomous vehicle maneuvering, a proposed algorithm using co-aligned intensity and range images, and simulation result. Chapter 7 contains the conclusions and proposed future work as a closing remark.

CHAPTER 2

MATHEMATICAL MORPHOLOGY

The term *morphology* refers to the study of shapes and structures from a general scientific perspective. Also, it can be interpreted as shape study using mathematical set theory. In image processing, morphology is the name of a specific methodology for analyzing the geometric structure inherent within an image. The morphological filter, which can be constructed on the basis of the underlying morphological operations, are more suitable for shape analysis than the standard linear filters since the latter sometimes distort the underlying geometric form of the image.

Some of the salient points regarding the morphological approach are as follows [10]:

1. Morphological operations provide for the systematic alteration of the geometric content of an image while maintaining the stability of the important geometric characteristics.
2. There exists a well-developed morphological algebra that can be employed for representation and optimization.
3. It is possible to express digital algorithms in terms of a very small class of primitive morphological operations.
4. There exist rigorous representation theorems by means of which one can obtain the expression of morphological filters in terms of the primitive morphological operations.

In general, morphological operators transform the original image into another image through the interaction with the other image of a certain shape and size, which is known as the structuring element. Geometric features of the images that are similar in shape and size to the structuring element are preserved, while other features are suppressed. Therefore, morphological operations can simplify the image data, preserving their shape characteristics and eliminate irrelevancies. In view of applications, morphological operations can be employed for many purposes, including edge detection, segmentation, and enhancement of images.

This chapter begins with binary morphology that is based on the set theory. Then, grayscale morphology can be regarded as the extension of binary morphology to a three-dimensional space since a grayscale image can be considered as a set of points in 3D space. The basic geometric characteristics of the primitive morphology operators are introduced in this chapter. A systematic introduction of theoretical foundations of mathematical morphology, its main image operations, and their applications can be found in [11], [10] and [12].

Mathematical morphology defined in a Euclidean setting is called Euclidean morphology and that defined in a digital setting is called digital morphology. In general, their relationship is akin to that between continuous signal processing and digital signal processing. The actual implementation of morphological operators will be in the digital setting, so in this thesis focusing on digital image, we only consider the digital morphological setting.

2.1 Binary Morphology

The theoretical foundation of binary mathematical morphology is set theory. In binary images, those points in the set are called the 'foreground' and those in the

complement set are called the 'background'.

Besides dealing with the usual set-theoretic operations of union and intersection, morphology depends heavily on the translation operation. For convenience, '$\bigcup$' denotes the set-union, '$\bigcap$' denotes set-intersection and '+' inside the set notation refers to vector addition in the following equations. We need two general definitions that are used extensively to extend morphological operations.

Definition: Reflection

Given an image B, the reflection of set B, denoted $\hat{B}$*, is defined as*

$$\hat{B} = \{w \mid w = -b, \text{ for } b \in B\}.$$

Actually, this operation has same effect as rotating image 180° about its origin.

Definition: Translation

Given an image A, the translation of A by the point x, denoted by A_x*, is defined by*

$$A_x = \{a + x \mid a \in A\}.$$

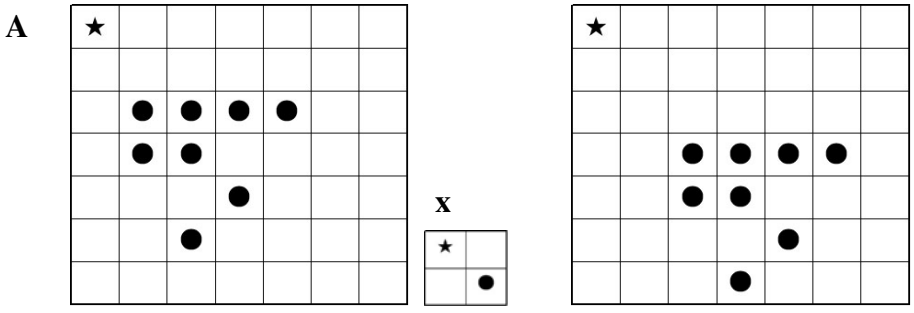


Figure 2.1: Illustration of the Translation Operation on Digital Setting

Figure 2.1 shows an example of the translation operation in digital space. The symbol ★ denotes the origin of each image set. The set A is an image in Z^2, with {(1,2),(2,2),(3,2),(4,2),(1,3),(2,3),(3,4),(2,5)} and x is a vector of (1,1). Translation of A by x results in {(2,3),(3,3),(4,3),(5,3),(2,4),(3,4),(4,5),(3,6)}. The result is described in the right part of Figure 2.1.

2.1.1 Structuring Element

Before continuing, we should describe what a structuring element is in morphological operations. A structuring element is a small image that is overlapped on input image to compute a certain definition. The basic operations of binary and also gray-scale images depend on what structuring elements are used. In this section, only those for binary morphology are considered. The gray-scale case will be described later. Figure 2.3 contains some examples that are used commonly for binary images.

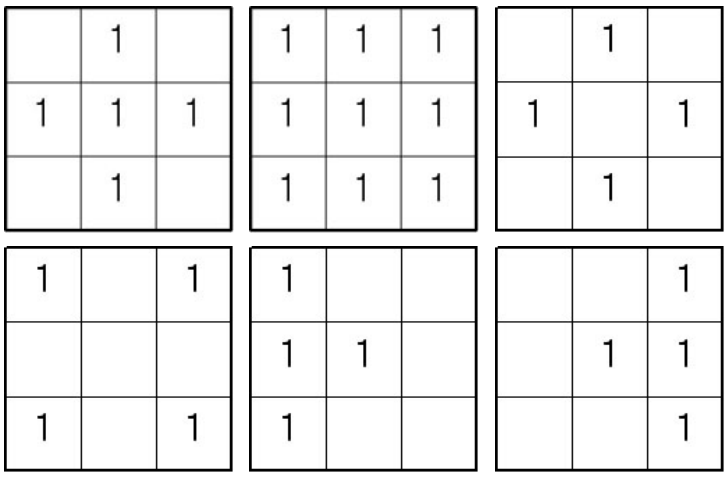


Figure 2.2: Examples of 3x3 Structuring Element

In these cases, all origins of structuring elements are located on the center. Depending on the shape of structuring element, the origin could be on a different location. The pixels marked with '1' are the points that should be considered during any binary

morphological operations.

2.1.2 Binary Dilation

Definition: Binary Dilation

With A and B as sets in Z^2, the dilation of A by B (usually A is an image and B is the structuring element), denoted by $A \oplus B$, is defined as

$$A \oplus B = \{z \in Z^2 \mid z = a + b, \text{ for some } a \in A \text{ and } b \in B\}.$$

It can be shown that dilation is equivalent to a union of translation of the original image with respect to the structuring element:

$$A \oplus B = \bigcup_{b \in B} (A)_b .$$

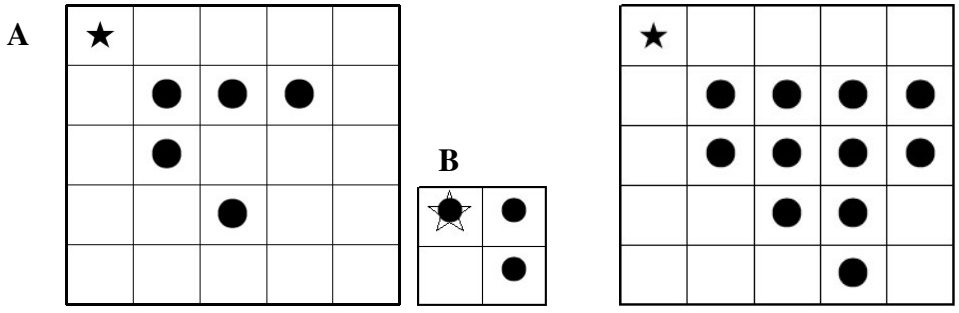


Figure 2.3: Illustration of Binary Dilation on Digital Setting

Dilation is found by placing the center of the template over each of the foreground pixels of the original image and then taking the union of all the resulting copies of the structuring element, produced by using the translation. From Figure 2.3, it is clear how dilation modifies the original image with respect to the shape of the structuring element.

Dilation generally has an effect of expanding an image; so consequently, small holes inside foreground can be filled.

In another sense, dilation can be a morphological operation on a binary image defined as:

$$A \oplus B = \{ z \mid \left(\hat{B} \right)_z \bigcap A \neq \phi \} .$$

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 is the set of all displacements, z, such that B and A overlap by at least one element. Based on this interpretation, the equation above may be written as

$$A \oplus B = \{ z \mid [\left(\hat{B} \right)_z \bigcap A] \subseteq A \} .$$

Although dilation is based on set operations, whereas convolution is based on arithmetic operations, the basic process -- "flipping" B about its origin and successively "displacing" it so that it slides over set (image) A -- is analogous to the convolution process. Even though dilation of an image A by structuring element B can be defined in several ways, all definitions have the same meaning and results in the same output. Figure 2.4 illustrates the dilation operation using a binary image. The original image is dilated with an 11x11 'disk' type structuring element.

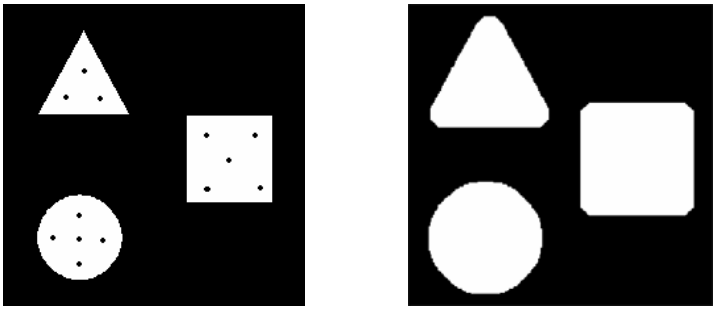


Figure 2.4: Binary Dilation Example

2.1.3 Binary Erosion

Definitions: Binary Erosion

Erosion of a binary image A by structuring element B, denoted by $A \Theta B$, is defined as

$$A \Theta B = \{ z \mid z + b \in A, \ \forall b \in B \} .$$

Whereas dilation can be represented as a union of translates, erosion can be represented as an intersection of the negative translates. So, the given definition of erosion above can be redefined as

$$A \Theta B = \bigcap_{b \in B} A_{-b}$$

where '–b' is the scalar multiple of the vector b by -1.

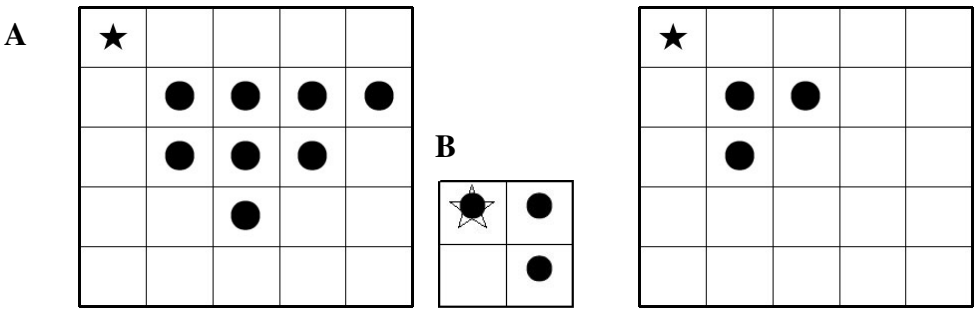


Figure 2.5: Illustration of Binary Erosion on Digital Setting

The erosion of the original image by the structuring element can be described intuitively by template translation as seen in the dilation process. Erosion shrinks the original image and eliminates small enough peaks (Note: the terms 'expand' for dilation and 'shrink' for erosion refer to the effects on the foreground). Figure 2.6 clearly illustrates these effects. The original image is eroded with 7x7 disk-shape structuring element.

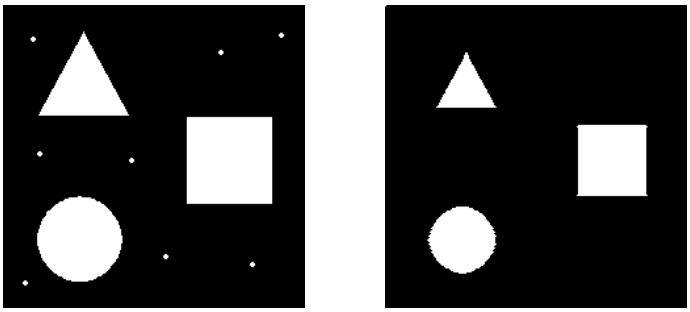


Figure 2.6: Binary Erosion Example

2.1.4 Binary opening

Definition: Binary Opening

The opening of a binary image A by the structuring element B, denoted by $A \circ B$, is defined as

$$A \circ B = (A \Theta B) \oplus B .$$

From the definition, the original image A is first eroded and then dilated by the same structuring element B. In terms of set theory, this opening process can also be defined as

$$A \circ B = \bigcup \{ (B)_z \mid (B)_z \subseteq A \} .$$

The whole procedure of opening can be interpreted as "rolling the structuring element about the inside boundary of the image".

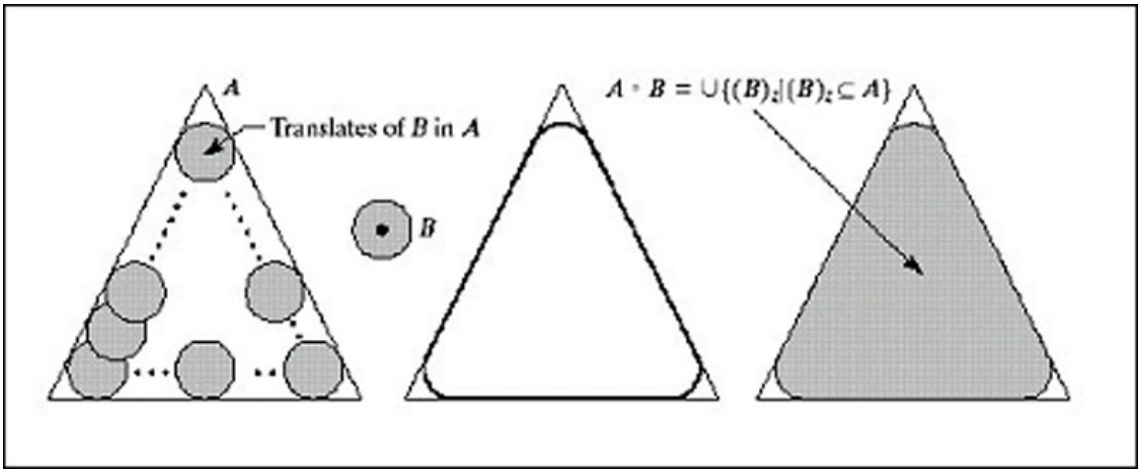


Figure 2.7: Illustration of Binary Opening Process [13]

The effects of the opening process on the original image are smoothing, reducing noise from quantization or the sensor and pruning extraneous structures. These effects result from the fact that the structuring element cannot fit into the regions. Therefore, it can be said that the result of the opening process heavily depends on the shape of structuring elements. Figure 2.8 presents an example of the opening process.

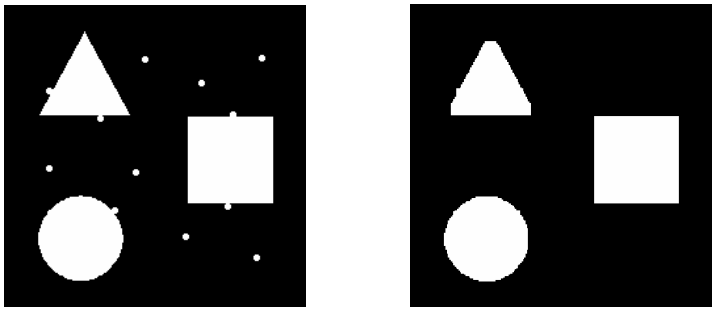


Figure 2.8: Binary Opening Example

The effects of the opening mentioned before are clearly shown in the Figure 2.8. The vortices of the triangle foreground have been cut out because the image is “opened” with ‘square’ type structuring element, whereas those of the square are preserved.

2.1.5 Binary closing

Definitions: Binary Closing

Closing of a binary image A by a structuring element B, denoted by $A \bullet B$*, is defined as*

$$A \bullet B = (A \oplus B) \Theta B .$$

In the closing operation, dilation and erosion are applied successively in that order. Note that this order is reversed for the opening process.

In another aspect, the closing process on a binary image can be defined as:

z is an element of $A \bullet B$ *if and only if* $(B + y) \bigcap A \neq \phi$*, for any translate (B+y) containing z.*

The closing operation can be described as in Figure 2.9 as "rolling the structuring element on the outer boundary of the image."

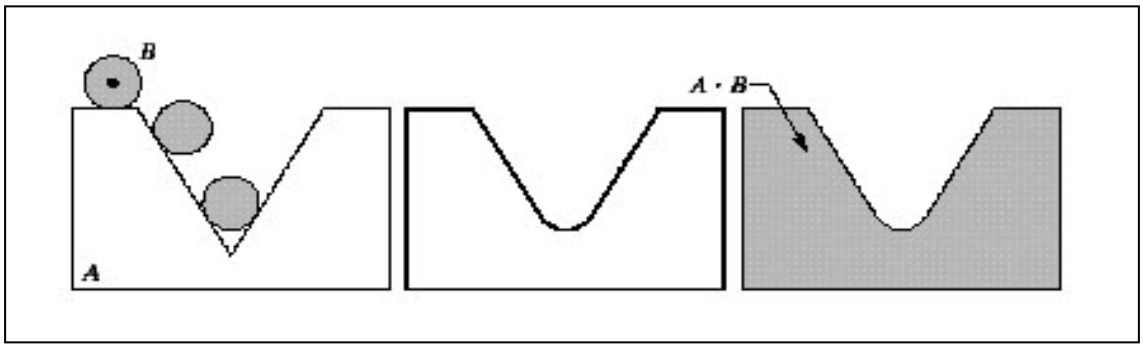


Figure 2.9: Illustration of Binary Closing Process [13]

The closing process has the effect of filling small holes in the original image, smoothing as the opening process does, and filling up the bay in the foreground. Sometimes, it is said that the closing has an effect of clustering each spatial point to be

connected. Figure 2.10 is an example showing how the closing operation works.

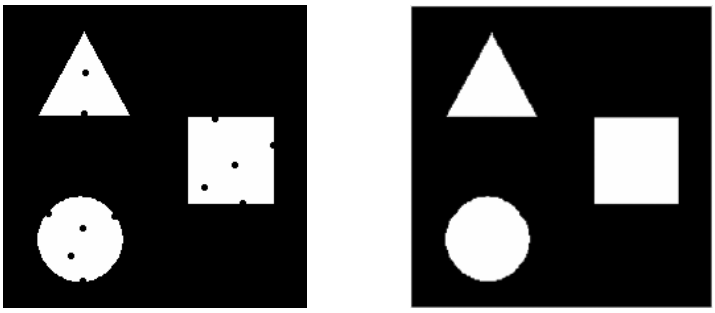


Figure 2.10: Binary Closing Example

2.2 Gray-scale Morphology

In this section, the binary morphology is extended to the gray-scale case. Through the discussions below, we will see that the key issue is to use the 'Maximum' and 'Minimum' functions to define gray-scale morphological operators. Using these concepts, gray-scale morphology can be easily extended from binary morphology with the same concept. The differences between binary and gray-scale morphology results from the definitions of dilation and erosion because other operations basically depend on these. Except for these definitions, gray-scale morphology is fairly similar to the binary case. Hence, in this section, definitions for gray-scale dilation and erosion as well as some of examples for gray-scale morphological operations are merely given. Also, some expansions of the morphological procedure are mentioned at the end.

2.2.1 Gray-scale Dilation and Erosion

Before discussing basic gray-scale morphological operations, it should be noted that the structuring elements of the gray-scale morphological operations could have the

same domains as those in binary morphology. However, as will be seen in the definitions below, a gray-scale structuring element has certain values ('b's') instead of having only position value '1' or '0' showing its domain. A grayscale image can be considered as a three-dimensional set where the first two elements are the x and y coordinates of a pixel and the third element is gray-scale value. It can be also applied to the gray-scale structuring element. With this concept, gray-scale dilation can be defined as follows

Definition: Gray-scale Dilation

Gray-scale dilation of f by b, denoted by $f \oplus b$, is defined as

$$(f \oplus b)(s,t) = \max\{\ f(s-x,t-y) + b(x,y) \mid (s-x),(t-y) \in D_f; (s,y) \in D_b\}.$$

where D_f and D_b are the domains of f and b, respectively.

Similarly gray-scale erosion can be defined as an extension of binary erosion.

Definition: Gray-scale Erosion

Gray-scale erosion, denoted by $f \Theta b$, is defined as

$$(f \Theta b)(s,t) = \min\{\ f(s+x,t+y) - b(x,y) \mid (s+x),(t+y) \in D_f; (s,y) \in D_b\}.$$

where D_f and D_b are the domains of each image or function.

Specific concepts and operation procedures are already explained in binary morphology.

Gray-scale dilation and erosion are duals with respect to function completion and reflection. That is, the relation between these can be expressed as

$$(f \Theta\ b)^c(s,t) = (f^c \oplus \hat{b})(s,t)$$

where $f^c = -f(x,y)$ and $\hat{b} = b(-x,-y)$.

The minimum operator will interrogate a neighborhood with a certain domain and select the smallest pixel value to become the output value. This has the effect of causing the bright areas of an image to shrink or erode. Similarly, grayscale dilation is performed by using the maximum operator to select the greatest value in a neighborhood. Figure 2.11 shows a simple image and its dilation and erosion with a "flat-top" structuring element. The term 'flat-top' refers to the fact that the values (b's) of the structuring element are all zero in a certain domain. In this example, a disk-type structuring element is applied.

a) original b) dilation c) erosion

Figure 2.11: An Example of Gray-scale Dilation and Erosion

2.2.2 Gray-scale Opening and Closing

Gray-scale opening and closing are defined below in a similar manner as the binary case. The only difference is, when the operations are carried out, these opening and closing operations use gray-scale dilation and erosion described in the previous section.

The effect of gray-scale closing and opening is shown in Figure 2.12. As binary morphological operations do, gray-scale opening is anti-extensive and gray-scale closing is extensive. Both operations make an original image smooth along to the nature of minimum

and maximum functions. Also, both operations have 'increasing', 'idempotent' properties.

a) original b) opening c) closing

Figure 2.12: An Example of Gray-scale Opening and Closing

2.3 Some Applications of Morphology

Although there are some applications requiring basic gray-scale morphological operations, most applications of morphology are developed for binary images. A list of binary morphological applications follows. The reader is referred to [13] for details of the individual applications.

1. Boundary extraction
2. Region filling
3. Extraction of connected components
4. Convex Hull
5. Thinning
6. Thickening
7. Skeletons
8. Pruning

All these applications are carried out by applying a series of basic operations with different type of structuring elements. For gray-scale morphological case, it can be

expanded to

1. Smoothing
2. Morphological gradient
3. Top-hat, Bottom-hat Transformations
4. Textual segmentation
5. Granulometry

In this thesis, we focus on how to use morphological gradient and how to improve that for image segmentation. The details of morphological gradient will be covered in Chapter 4 as a pre-processing stage of watersheds transformation.

CHAPTER 3

WATERSHEDS TRANSFORMATION IN DIGITAL SPACE

In this chapter, the historical background of watersheds and the evolution of the algorithm are introduced first. Later, the fast watersheds algorithm is reviewed and the key issues used to make the algorithm fast are especially pointed out and explained generally. Finally, the some potential problems with the watersheds transformation are described.

3.1 Introduction to the Concept of Watersheds.

The watersheds concept is one of the classic tools in the field of topography. It is the line that determines where a drop of water will fall into a particular region. In image processing, especially mathematical morphology, grayscale images are considered as topographic relieves. In the topographic representation of given image I, the intensity value of each pixel stands for the elevation as this point.

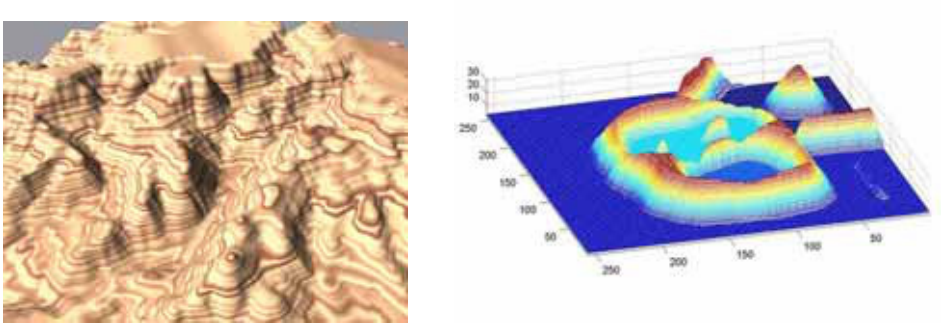


Figure 3.1: Topographical Map and Gray-scale Image in 3D Representation

Naturally, the first algorithm for computing watersheds is found in the field of topography. Topographic surfaces are numerically handled through digital elevation models (DEM's). These are arrays of numbers that represent the spatial distribution of terrain altitudes. The most commonly used data structure for DEM's is the regular square grid in which available elevations are equally spaced in two orthogonal directions. These restrictions are very similar to gray scale images in digital spaces.

Unlike the typical morphological filters, the watersheds transformation is not composed of the primitive morphological operations. The initial concept of the watersheds transformation as a morphological tool was introduced by H. Digabel and C. Lantuéjoul in [7]. Later, a joint work of C. Lantuéjoul and S. Beucher led to the 'inversion" of this original algorithm in order to extend it to the more general framework of grayscale images. Later, watersheds were studied by many other researchers and used in numerous grayscale segmentation problems. In this thesis, the efficient algorithm for watersheds suggested by Luc Vincent and Pierre Soille is reviewed briefly and used throughout the entire simulation.

As an interpretation in topography, the watershed can be imagined as the high mountain that separates two regions. Each region has its own minimum and, if a drop of water falls on one side of the watershed, it will reach the minimum of the regions. The regions that the watershed separates are called catchment basins.

3.2 Watersheds Transformation Algorithm

The algorithm introduced by Luc Vincent and Pierre Soille is based on the concept of "*immersion*" [6]. Each local minima of a gray-scale image I which can be regarded as a surface has a hole and the surface is immersed out into water. Then, starting from the minima of lowest intensity value, the water will progressively fill up different catchment basins of image (surface) I. Conceptually, the algorithm then builds a dam to avoid a situation that the water coming from two or more different local minima would be merged. At the end of this immersion process, each

local minimum is totally enclosed by dams corresponding to watersheds of image (surface) *I*. Figure 3.2 shows this procedure graphically.

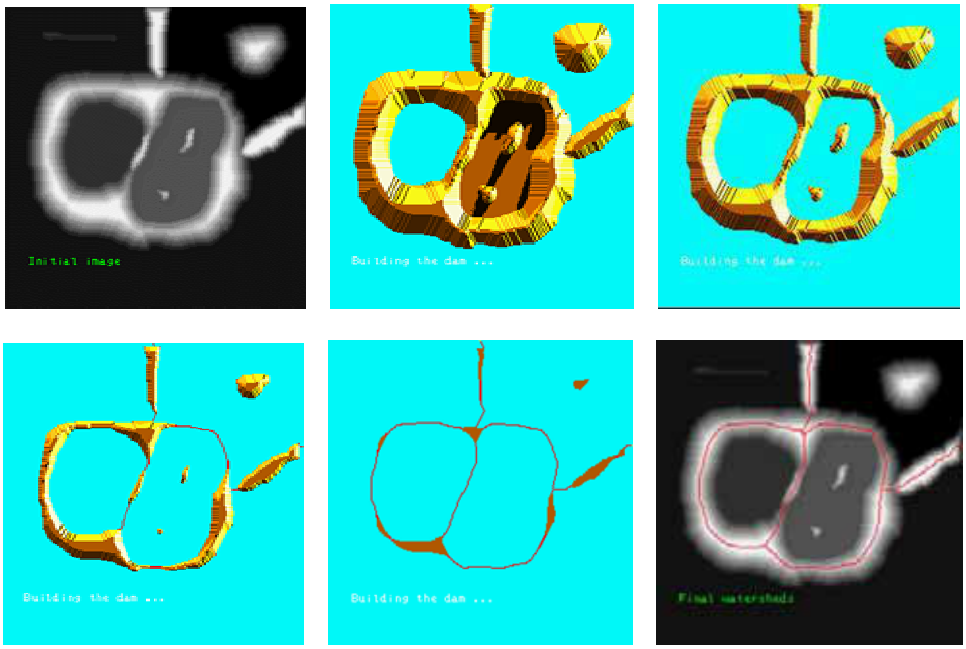


Figure 3.2: The Immersion Procedure [17]

This informal analysis of the immersion process can be established mathematically by the definition of '*geodesic distance*' and '*geodesic influence zone*'. 'Geodesic influence zone' deals with the expansion of the plateau from each local minimum for watersheds transformation.

Definition: Geodesic Distance

The geodesic distance $d_A(x,y)$ between two pixels x and y in A is the infimum (greatest lower bound of given set) length of the paths which join x and y and are totally included in A.

Definition: Geodesic Influence Zone

Suppose A contains a set B consisting of several connected components B_1, B_2 ,… B_k. The geodesic influence $iz_A(B_i)$ of a connected component B_i of B in A is the locus of the points of A whose geodesic distance to B_i is smaller than their geodesic distance to any other component of B.

It can be expressed as

$$iz_A(B_i) = \{ p \in A,\ \forall j \in [1,k]/\{i\}, d_A(p,B_i) \}.$$

The immersion process and its results along with catchment basins and watersheds can be described in terms of geodesic influence zone.

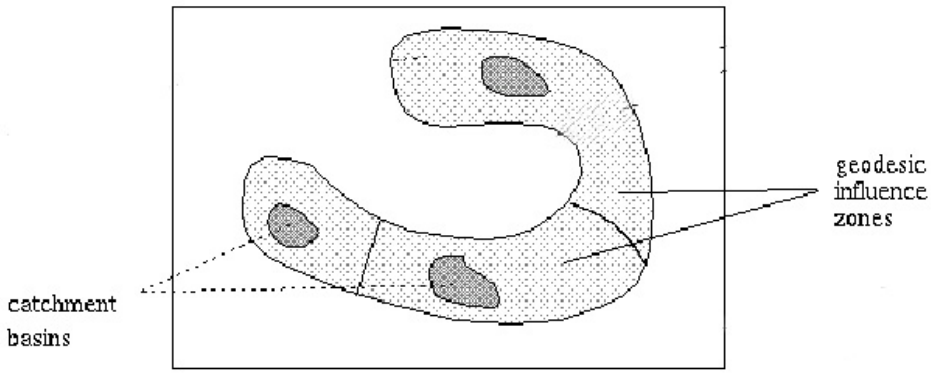


Figure 3.3: Illustration of Geodesic Influence Zone

Definition: Catchment Basins and Watersheds by Immersion

The set of the catchment basins of the gray-scale image I is equal to the set X_{hmax} obtained after the following recursion:

1. $X_{h_{\min}} = T_{h_{\min}}(I),\ T_h(I) = \{ p \in D_I,\ I(p) \le h \}.$
2. $\forall h \in [h_{\min,} h_{\max} - 1],\ X_{h+1} = \min_{h+1} \bigcup iz_{T_{h+1}(I)}(X_h).$

The immersion procedure is done in the recursion, and the watersheds of I correspond to the set of the points of D_I which do not belong to any catchment basin.

From these basic concepts and primitive algorithm, Luc and Pierre improved the immersion-based algorithm for watersheds extraction. The algorithm consists of two important

steps, sorting and flooding and the queue structure is used to make computation time shorter. A mathematical dilation process is used to find new local minimum candidates and to find the geodesic influence zone in flooding. C-friendly pseudo code is attached as Appendix A to show the basic algorithm. Fortunately, Mathworks Matlab™ has a built-in function of the watersheds transformation ('watershed') in its Image Processing Toolbox using the same algorithm. The routine takes a binary or gray-scale image as an input and makes a same size image as an output, labeling different regions with different numbers. Watersheds are denoted as '0' regions. Figure 3.3 is an example of watersheds transformation carried out on the 'Lena' gray-scale image.

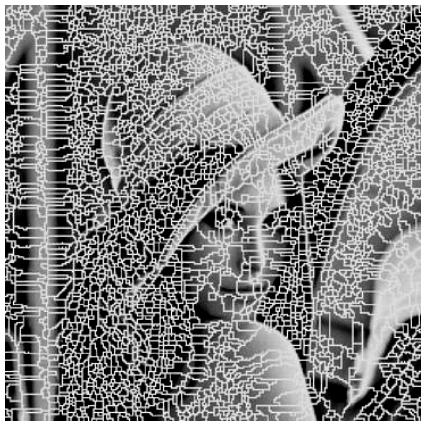


Figure 3.4: An Example of the Watersheds transformation on Gray-scale Image

3.3 Embedded Problems

The watersheds transformation is an effective method for extracting out continuous boundaries of each region. However, applying it to the original image can cause undesired results. In this section, we will go over these problems with some simple examples and introduce some methods to improve the results of the watersheds transformation.

3.3.1 Oversegmentation

The watersheds transformation makes a number of regions as an output. For example, a

human can clearly see a woman with background in Figure 3.4. This is because humans are capable of understanding the 'semantics' of a given scene; however, this image has 2976 different regions after the watersheds transformation. This oversegmentaion problem comes mostly from the noise and quantization error. To eliminate the effect of local minima from noise or quantization error on the final results, first, the gradient of the original image is computed as a pre-processing and then the watersheds transformation is applied on the gradient of image. Noise or quantization error has a quite different value relative to its neighbor; hence, it shows a high gradient value. So, if the watersheds transformation is applied to the gradient image, this high value is no longer a 'local minimum'. Another approach to eliminate noise and quantization error effects is to apply 'region merging' algorithm as a post-processing. A large number of regions are merged until the output meets a given criteria which can be the number of regions or a dissimilarity value between homogeneous regions.

3.3.2 Ambiguous Boundary on Homogeneous Regions.

Another reason why the watersheds transformation is applied to the gradient image is because the watersheds transformation makes ambiguous boundaries occasionally. Figure 3.5 shows a simple case with a vague boundary. As interpreted, it consists of 3 different homogeneous regions. If the watersheds transformation is applied directly to the original image, then the output is just 2 different regions. The only marked boundary is made across the middle homogeneous regions. It does not match the original boundaries. However, if the transformation is applied to the gradient of the original image, it can extract out proper boundaries and recognize 3 different homogeneous regions.

In Chapter 4, as a pre-processing stage, the morphological gradient compared to the conventional gradient-like Sobel, Laplacian operators- is reviewed and extended to multi-scale to find out more detail about the gradient. For the post-processing, a region merging algorithm is described in Chapter 5.

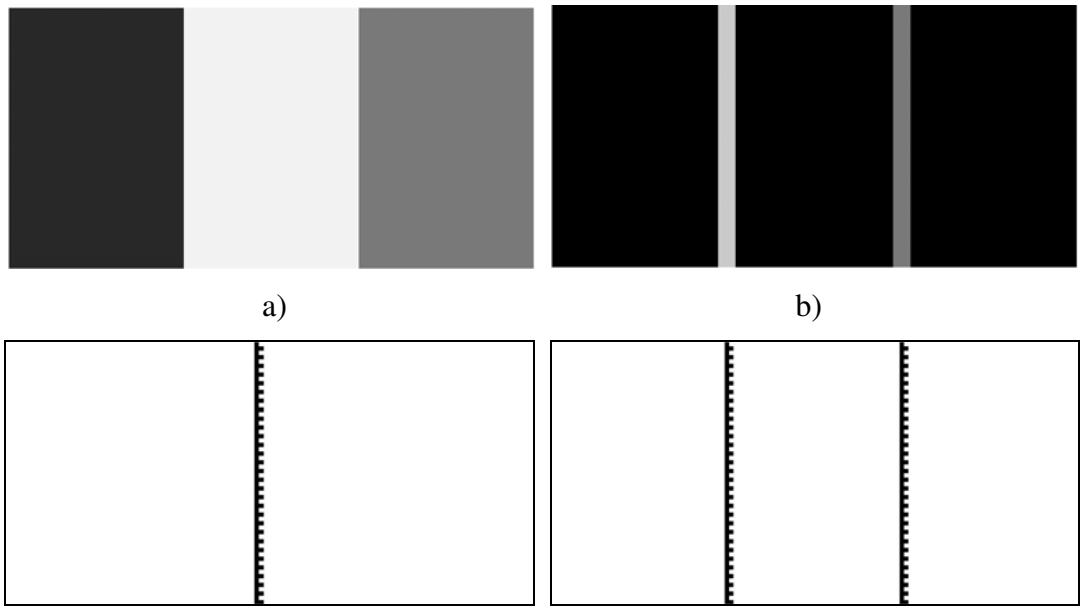


Figure 3.5: Simple Illustration of Ambiguous Boundary Extraction:
a) original, b) gradient, c) WS on original, d) WS on gradient

CHAPTER 4

MORPHOLOGICAL GRADIENT OPERATORS

Image edge detection is a basic tool in image segmentation since edges carry valuable features of the image. Edges in an image are formed due to variations of illumination in the scene. Hence, the conventional approach to edge detection is composed of "gradient calculation" and "thresholding". For the first step, the original image is transformed into a gradient image which represents the edge strength of each pixel. A threshold is then applied to classify each pixel to the edge point or non-edge point. Traditionally, the gradient image can be obtained by means of first-order differential operators or a Laplacian operator which can enhance the spatial intensity changes in the image. Morphological edge detectors have also been proposed for their robustness under noisy conditions and some of them are discussed in the following section.

4.1 Conventional Gradients

Traditional methods to find the gradient of an image typically apply a first-order differential operation on the original image. This operation is similar to linear filtering which consists of scanning the original image with certain masks. The sizes and values of the mask are derived from standard mathematical expressions and approximated with ad hoc procedures to digital settings:

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}.$$

The following are examples of well-known gradient operators along with a typical output. Note that Laplacian operator is derived from the second-order differential

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}.$$

−1	−1	−1
0	0	0
1	1	1

−1	0	1
−1	0	1
−1	0	1

a) Gx mask, Gy mask, and the Gradient

Figure 4.1: Conventional Gradient Operators on ‘Lena’:
a) Prewitt, b) Sobel, c) Laplacian

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

b) Gx mask, Gy mask, and the Gradient

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

c) 4-neighbor, 8-neighbor, and Output

Figure 4.1 Continued

For the case using first-order operation, the masks in Figure 4.1 are designed to detect vertical and horizontal edges. Different patterns of the mask should be applied to detect edges that appear diagonally in the image.

0	1	1
−1	0	1
−1	−1	0

−1	−1	0
−1	0	1
0	1	1

0	1	2
−1	0	1
−2	−1	0

−2	−1	0
−1	0	1
0	1	2

Figure 4.2: Prewitt, Sobel Mask for Diagonal Edges

First-order and second order gradient operators (conventional gradients) are convenient to use and relatively fast to calculate the gradient. The procedure consists of overlapping the original image with certain masks and adding those together to find out the gradient value at a point and then repeating these through the whole image. These methods perform strongly to detect points, line, or edges of objects in the situation that the given image does not contain noise. However, if there is noise in the image set, these operators consider the noise pixels as points because they are too sensitive. Naturally, digital images have noise due to imperfections in image acquiring sensors like CCD and CMOS. Especially, the noise in the image is generated more where light is not controlled

like the pictures taken by autonomous vehicle (AV).

4.2 Morphological Gradients

The morphological gradient is based on the concept of residues which make difference between the transformations of filters. Some simple morphological gradient operators are illustrated as follows [1]:

Gradient by Erosion: $Ge(f) = f - (f \Theta B)$.

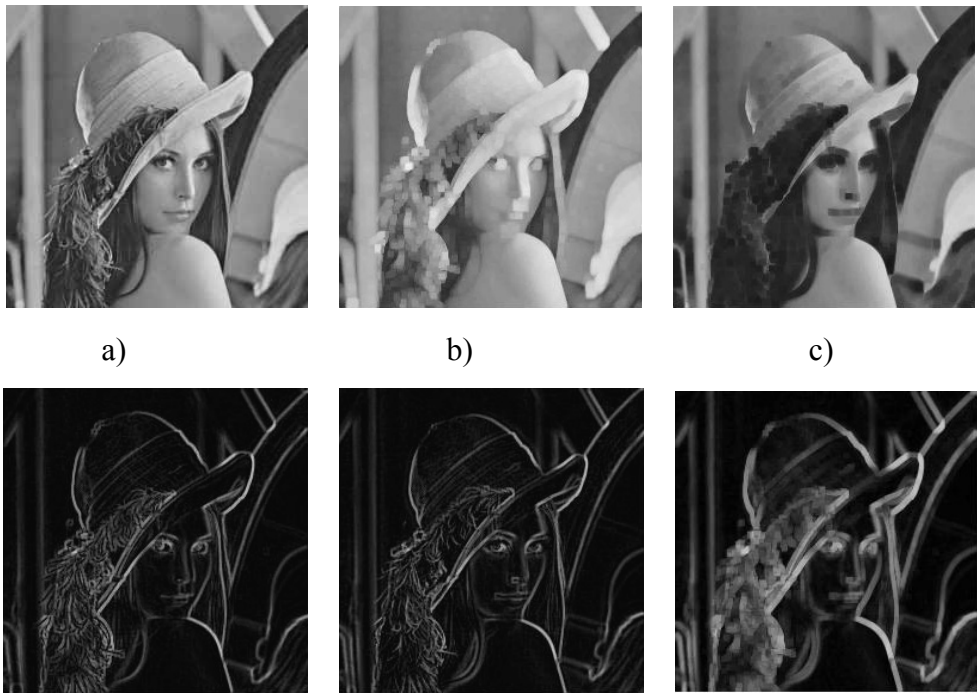
Gradient by Dilation: $Gd(f) = (f \oplus B) - f$.

Morphological Gradient: $G(f) = (f \oplus B) - (f \Theta B)$.

If B is chosen as the rod structure element with flat top whose domain is the origin and its 4-neighbor, then gradient by erosion and dilation are also called erosion residue edge detector and dilation residue edge detector respectively. Furthermore, these operations can be reduced to the following mathematical expressions [2]:

$$\begin{aligned} Ge(r,c) &= f(r,c) - (f \Theta b)(r,c) \\ &= f(r,c) - [\min_{(i,j)\in D_{rod}} (f(r-i, c-j))]. \end{aligned}$$

$$\begin{aligned} Gd(r,c) &= (f \oplus b)(r,c) - f(r,c) \\ &= [\max_{(i,j)\in D_{rod}} (f(r-i, c-j))] - f(r,c). \end{aligned}$$



a) b) c)

d) e) f)

Figure 4.3: Basic Morphological Gradients:
a) original, b) dilated, c) eroded, d,e) gradient by dilation, erosion, f) morph. gradient

The basic morphological residue gradients are still a little sensitive to quantization error or noise and are position biased. Edge strength is only given to that side of the edge which has the lower/higher value.

The first robust morphological edge detector was proposed by Lee, Haralick and Shapiro. Called blur-minimization edge detector, this method is noise insensitive and shows better performance detecting of step and ramp edges. More details and experimental results can be found in the original paper [3]. The main equation is

$$f_o(z) = \min\{f_{av}(z) - (f_{av} \Theta B_v)(z),\ (f_{av} \oplus B_v)(z) - f_{av}(z)\}$$

where $f_{av}(z)$ is the input image blurred with a 2D running mean filter, $f_o(z)$ is the output

image, and B is the structuring element which is a square with (2n+1)*(2n+1) pixels.

For greater efficiency, the mean filter blurring of the original image can be replaced with the α-trimmed mean filter.

Figure 4.4: Blur-minimization Edge Detector:
a) blurred image b) robust morphological gradient

4.3 Multiscale Edge Detectors

For greater robustness to noise, a multiscale gradient algorithm can be considered. The term ‘multicale’ means to analyze the image with structuring elements of different sizes. Intuitively, using the structuring element of smaller size may detect fine edges but become more sensitive to noise. On the other hand, the larger the size of the structuring element, the more noise can be removed but the edge becomes thicker. Hence, the combination of the morphological gradients in different scales is able not only to become insensitive to noise but also to extract various fineness of the edge.

In the paper [2], a multiscale morphological edge detector is introduced. Starting with some of the disadvantages of classical definition of morphological gradient, the improved erosion residue operator and improved dilation residue operator are defined as follows:

- **Improved Erosion Residue Operator**

$$G'_e(r,c) = \min\{f(r,c) - erosion_{D_{rod1}}(r,c),\ f(r,c) - erosion_D(r,c),\ G''_e(r,c)\}$$

where $G''_e(r,c)$ is defined as

$$G''_e(r,c) = \max\{|erosion_{D_1}(r,c) - erosion_{D_2}(r,c)|,\ |erosion_{D_3}(r,c) - erosion_{D_4}(r,c)|\}.$$

- **Improved Dilation Residue Operator**

$$G'_e(r,c) = \min\{dilation_{D_{rod1}}(r,c) - f(r,c),\ dilation_D(r,c) - f(r,c),\ G''_d(r,c)\}$$

where $G''_d(r,c)$ is defined as

$$G''_d(r,c) = \max\{|dilation_{D_1}(r,c) - dilation_{D_2}(r,c)|,\ |dilation_{D_3}(r,c) - dilation_{D_4}(r,c)|\}.$$

Recall that the D_i's are the structuring elements with flat top and their domains are defined as below. The reason for using different structuring elements is to consider the direction of the gradient of image.

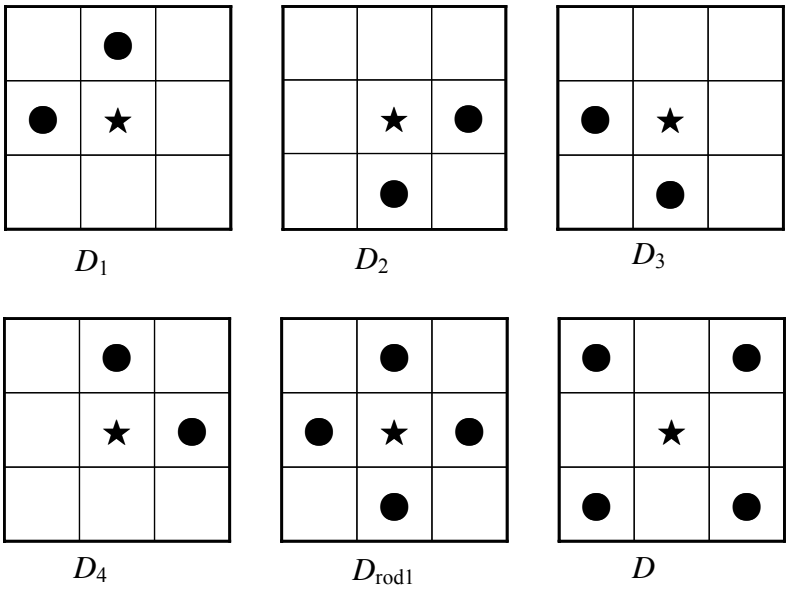


Figure 4.5: Domains of the Structuring Elements for Multiscale

Each structuring element is defined with radius 1. To extend the above operators to scale *n*, the structuring elements are modified in proportion to *n*. That is, nD_1, nD_2, nD_3, and nD_4 have the same domain as D_1, D_2, D_3 and D_4 but have size *n*. Similarly, D_{rodn} is defined as nD_{rodn} and $nD=nN_8/nN_4 \cup (0,0)$ (where N_8 means 8-neighbor and N_4 means 4-neighbor). Therefore, the improved dilation residue operator at scale *n* is defined as:

$$G_d^n(r,c) = \min\{dilation_{D_{rodn}}(rmc) - f(r,c),\ dilation_{nD}(r,c) - f(r,c),\ G_d^{n'}(r,c)\}$$

where $G_d^{n'}(r,c)$ is defined as

$$G_d^{n'}(r,c) = \max\{|\, dilation_{nD_1}(r,c) - dilation_{nD2}(r,c)\,|,\ |\, dilation_{nD3}(r,c) - dilation_{nD_4}(r,c)\,|\}.$$

The improved erosion residue operator at scale *n* can be defined in a similar way. In [2], the improved dilation residue operator is focused and it is mentioned that edge

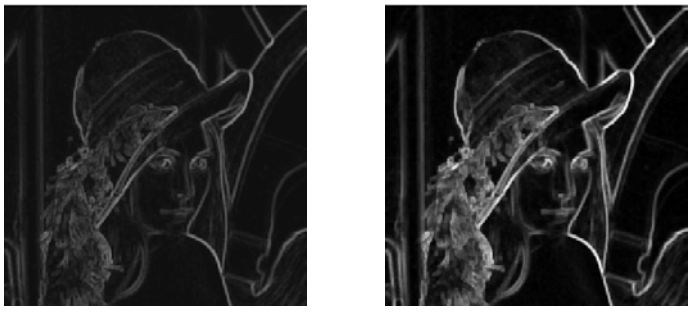
strength can be defined by the improved erosion residue operator or by sum of them.

After defining the edge image of each scale, the task turns to combining the edge strength. It is very intuitive to use the weighted pixel summation, i.e.,

$$f'(r,c) = \sum_{n=k}^{l} w_n f_n'(r,c)$$

where [k ,l] represents the range of scale and w_i's are respective weights of each scale. Simply, the weights can be equally distributed for all scale. Other kinds of combinations are also possible.

Since the edge strength of each pixel has been determined, each pixel can be directly thresholded to classify it into the categories of either edge or non-edge. In [2], the authors makes use of "Non-maximal suppression". The combined edge image appears to contain a long range of mountains and the true edge lies along its ridge (see Figure 4.6); hence, the watersheds transformation introduced in the previous section is suitable for extracting the ridge points in an edge image.



a) n=1 b) n=3

Figure 4.6: Multiscale Edge Detector at Scale

Based on the concepts stated previously, the applicable multiscale edge detector is

introduced in [4] to find the gradient of the image:

$$MG(f) = \frac{1}{n}\sum_{i=1}^{n}[((f \oplus B_i) - (f\ \Theta\ B_i))\ \Theta\ B_{i-1}]$$

where n is the scale and B_i denotes the group of square structuring elements whose sizes are (2i+1)*(2i+1) pixels. The choice of square structuring elements has the advantage of low computational cost. Other structuring elements like line segments of different directions can be used for specific applications.

CHAPTER 5

REGION MERGING

In Chapter 3, some embedded problems of the watersheds transformation were discussed, and a multiscale morphological gradient algorithm was introduced as a pre-processing stage to reduce the effect of noise and quantization error in Chapter 4. In this chapter, a region merging post-processing stage is explained to compensate for the over-segmentation problem. The most natural method to overcome the over-segmentation of watersheds transformation is to merge the small regions in a homogeneous region since they may possess certain homogeneous characteristics in intensity, texture or statistical properties.

There is a traditional method for image segmentation, called Split/Merging which was already mentioned in Chapter 1. The Split/Merging method takes an intensity image as an input and splits it into small grids usually using quadtree structure. Finally, the procedure merges small grids according to their statistical properties.

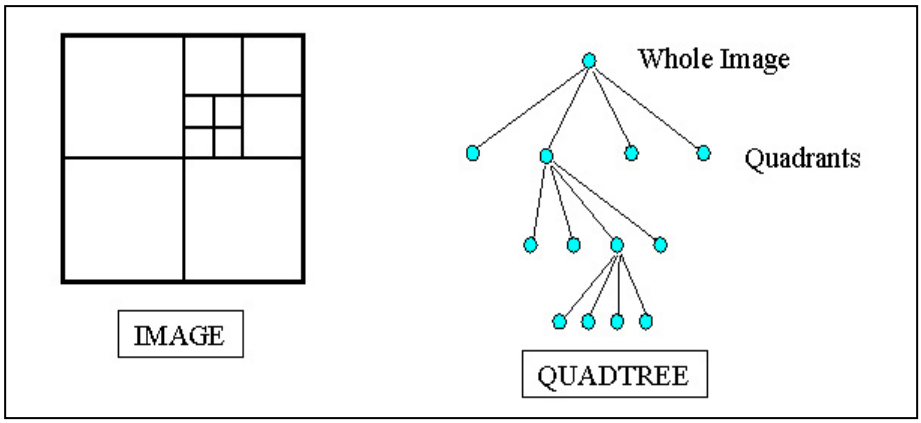


Figure 5.1: Quadtree Representation

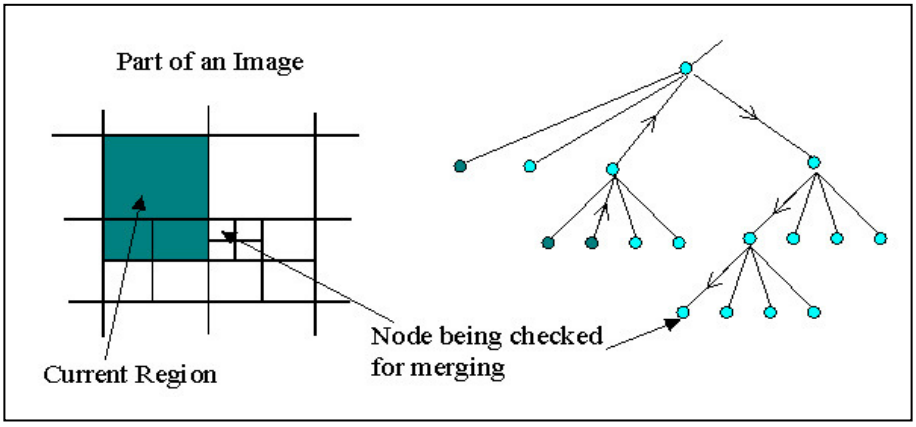


Figure 5.2: Split and Merging with Quadtrees

The region merging as post-processing for watersheds transformation takes a labeled image as input instead. This labeled image coincides with a quadtree of Split/Merging method. As explained in Chapter 3, the watersheds transformation algorithm processes the original image into a labeled image with boundary pixels; each label represents a different region. Two important keys for merging different regions together are:

1. If the regions are adjacent or not
2. How dissimilar/similar the regions are to each other.

The sections of this chapter contain the mathematical concepts and the implementations in a Matlab™ program about each key respectively.

5.1 Region Adjacency Graph

A simple label image is shown in Figure 5.3. It can be transformed into a Region Adjacency Graph (RAG), to indicate whether two regions are adjacent or not; only adjacent regions are connected with bars.

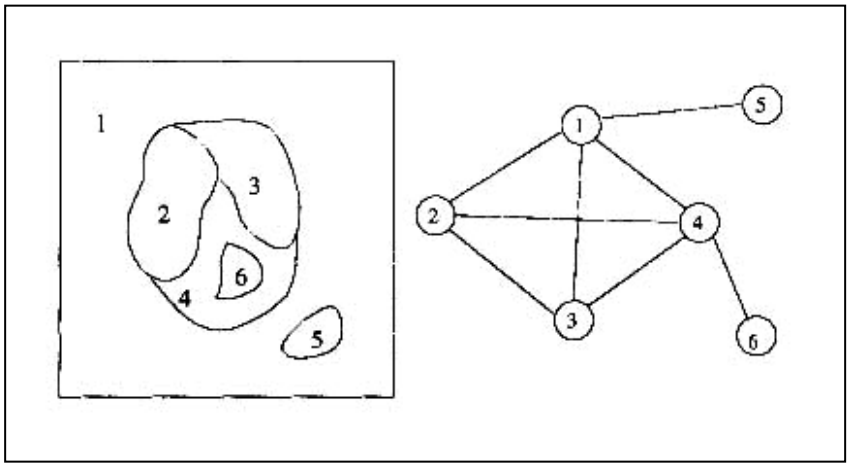


Figure 5.3: Simple Label Image and its RAG

Two regions that have minimum cost dissimilarity (a concept that will be explained in the next section) are merged together if the dissimilarity satisfies a given criteria. Figure 5.4 shows the merging step between regions a and b where it is assumed that those regions have minimum cost dissimilarity.

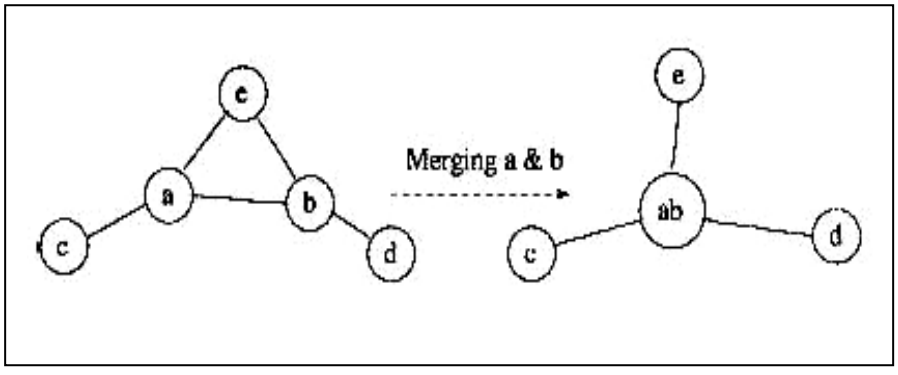


Figure 5.4: Merging Steps using RAG

This merging step is repeated until there is no pair of regions satisfying the dissimilarity condition or the number of regions in repetition is the same as a given number

of regions. Each merging step reduces the number of regions by 1. Based on conceptual analysis, the pseudo-code for region merging can be summarized. Given the RAG of the initial *K*-partition (*K*-RAG), the RAG of the suboptimal (*K*-*n*)-partition ((*K*-*n*)-RAG) is constructed by:

Input: RAG of the *K*-Partition (*K*-RAG)

Iteration: For *i*=0 to *n*-1

Find the minimum cost edge in the (*K*-*i*)-RAG

Merge the corresponding pair of regions to get the (*K*-*i*-1)-RAG

Update RAG and dissimilarity

Output: RAG of the (*K*-*n*)-partition ((*K*-*n*)-RAG)

When region merging is applied to a labeled image being implemented in a program language, the image can be processed by first, scanning the image with a 3x3 window and the comparing labels in the window. Then, the table containing the RAG information is built in matrix form. This table represents the status of adjacency with a flag (0=off, 1=on).

	Region_1					Region_6
Region_1	x	1	1	1	1	0
	1	x	1	1	0	0
	1	1	x	1	0	0
	1	1	1	x	0	1
	1	0	0	0	x	0
Region_6	0	0	0	1	0	x

Figure 5.5: RAG Table

The dissimilarity cost is calculated only for the region pairs that have their RAG flags on.

5.2 Region Dissimilarity Function

Similarity between two regions can be simply described by the difference of statistical properties like average, variation or both of intensity values for each region. In this thesis, the dissimilarity function defined as the equation below is used. This objective cost function is the square error of the piecewise constant approximation of the observed image, which yields a measure of the approximation accuracy and is defined over the space of partitions. If R^*_M is the optimal M-partition with respect to the squared error, then the optimal (M-1)-partition is generated by merging the pair of regions of R^*_M, which minimizes the dissimilarity function [14], [15]:

$$\delta(R_M^{*i}, R_M^{*j}) = \frac{\| R_M^{*i} \| \bullet \| R_M^{*j} \|}{\| R_M^{*i} \| + \| R_M^{*j} \|} [\mu(R_M^{*i}) - \mu(R_M^{*j})]^2 I(i, j).$$

where

$$I(i, j) = \begin{cases} 1, & \text{if regions } R_M^{*i} R_M^{*i} \text{ are adjacent} \\ +\infty, & \text{otherwise} \end{cases}.$$

According to the above formulation, the most similar pair of regions is the one minimizing square error.

The determination of the optimal number of segments K^* is performed by checking the value of $\delta(\bullet,\bullet)$. If δ is greater than a certain threshold, then the merging process is terminated. This threshold value can be obtained through hypothesis testing on noise distribution; however, the desired number of regions can be simply given to stop the merging process if the threshold value is not certain [16].

Figure 5.6 shows the effect of region merging. The original image is processed with the watersheds transformation and region merging is applied to reduce the number of regions. In this case, the program stops the merging process if the average intensity difference between the optimal pair of regions being merged is greater than 12.

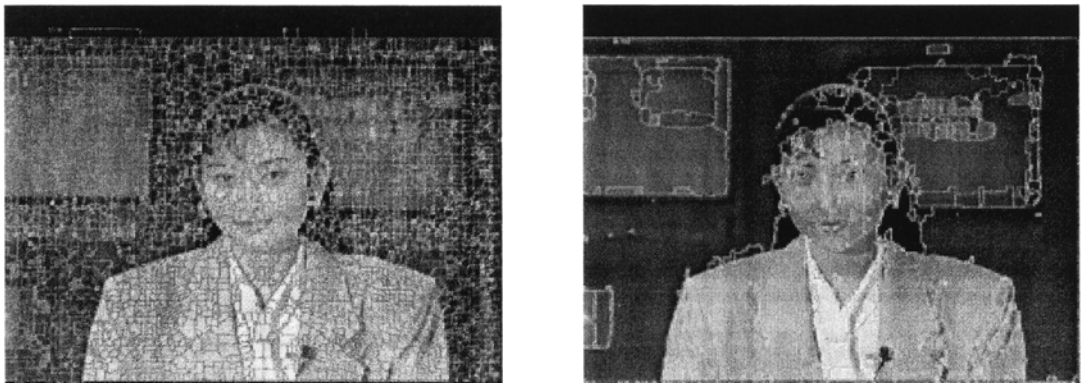


Figure 5.6: Region Merging of Watersheds on Akiyo

CHAPTER 6

APPLICATION AND SIMULATION:

IMAGE SEGMENTATION USING CO-ALIGNED IMAGES

6.1 Motivation

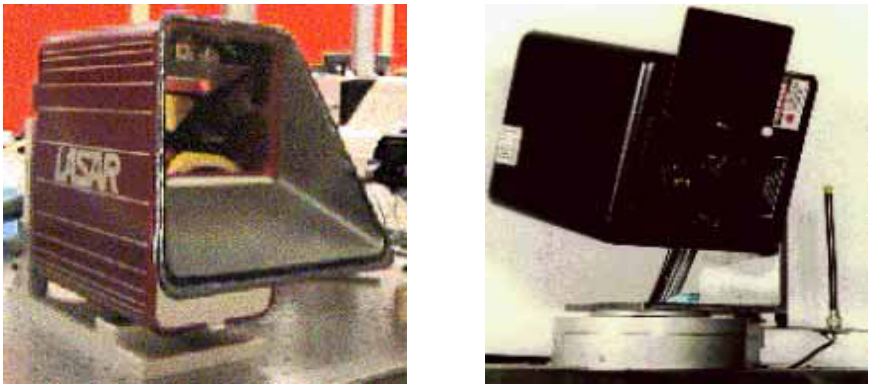
Research on autonomous vehicles (AV) is recently expanding in the robotics field. One of the key issues is the vision system of the AV. Vision sensors like intensity and infrared cameras acquire information about the AV's surroundings including objects, obstacles, and targets etc. As a primitive processing, the image segmentation takes an important role in image processing, which is a very crucial part of an AV system that allows the AV to maneuver around without problems. Also, the result of image segmentation has a tremendous effect on the following image processing steps like automatic target recognition and obstacle avoidance. The idea of this thesis is to use both intensity and range information to improve the segmentation of the given scene image into homogeneous regions. This procedure can improve the whole AV vision system because most of the image processing depends on the result of the image segmentation step.

6.2 Application: the Proposed System

6.2.1 Co-Aligned Images

An AV is usually equipped with intensity and range cameras. LADAR (LA(ser) D(etecting) A(nd) R(anging)) is a popular range sensor; however, it does not have very

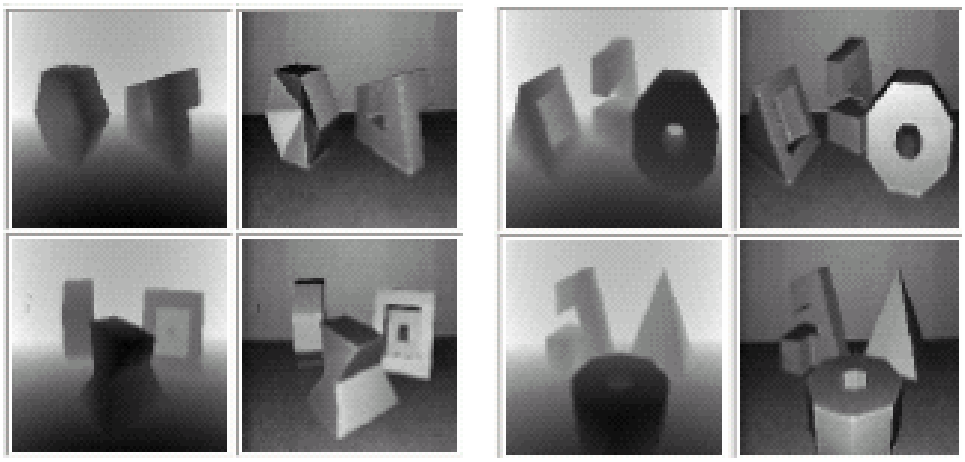
good resolution compared to an intensity camera. To obtain more detailed range data, a high-resolution range camera like ABW, K2T, Perceptron and Odetics can be used. While ABW and K2T range cameras recover depth from triangulation using structured light, Perceptron and Odetics range cameras do the same processing by computing the phase shift between the outgoing laser beam and its returned (bounded back) signal. Co-aligned images can be acquired from intensity and high-resolution range cameras. Two cameras can be arranged physically for acquiring a co-aligned image or the image can have same alignment using a transformation matrix with the given focal length, relative coordinates between cameras.



a) Perceptron b) Odetics

Figure 6.1: High Resolution Range Camera

An image data base from the CESAR Lab at Oak Ridge National Laboratory is used for simulations. Figure 6.2 shows examples of the co-aligned image data base.



range intensity range intensity

Figure 6.2: Examples of Co-aligned Image Data

6.2.2 Extraction Edge Information from Multiple Data.

Valuable edge information can be extracted from both intensity and range images using the morphological gradient. As reviewed in previous sections, the multiscale morphological gradient method is more reliable for edge detection when there is noise compared to conventional methods. Using only the intensity image or only the range image for edge detection, it is not enough to reveal all edges of a given scene. Figure 6.3 shows a simple example of hidden edges in an intensity image. Two objects are arranged dexterously, so that if viewed from the front of the objects and using only intensity image or only the range image, it is not possible to find the actual border between two objects. The gradient of the range image helps to reveal that edge. It gives an idea that compensation can be made with a combination of both.

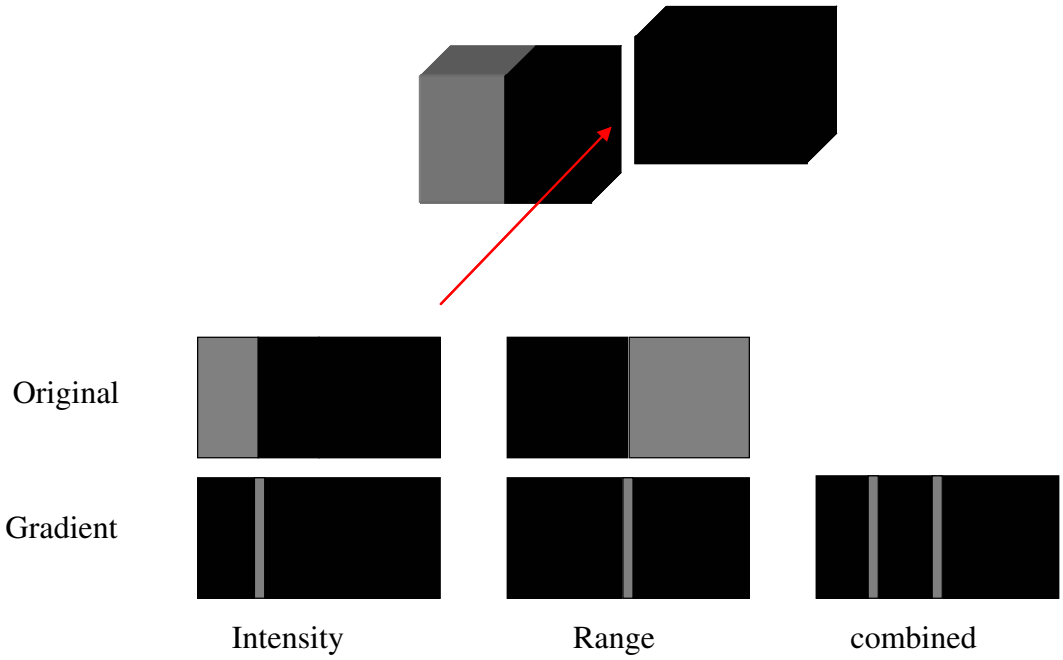


Figure 6.3: The Idea of Hidden Edge

6.2.3 Proposed Method

Figure 6.4 shows the system diagram of the proposed method. The contents of most of the boxes are covered in previous sections. Through the series of the process, the segmented image finally can be acquired. First of all, both images are read in and preprocessed and the multiscale morphological gradient is applied separately to each image. After that, the edge information from each process is combined using selected methods. For the combination of each edge information, two methods are tested in this simulation—linear average and maximum selection. Primitive segmentation is calculated by applying the watersheds transformation. The final results are produced through region merging using the algorithm reviewed in Chapter 5. For the merging criterion, the dissimilarity of the average intensity value of a segment is used.

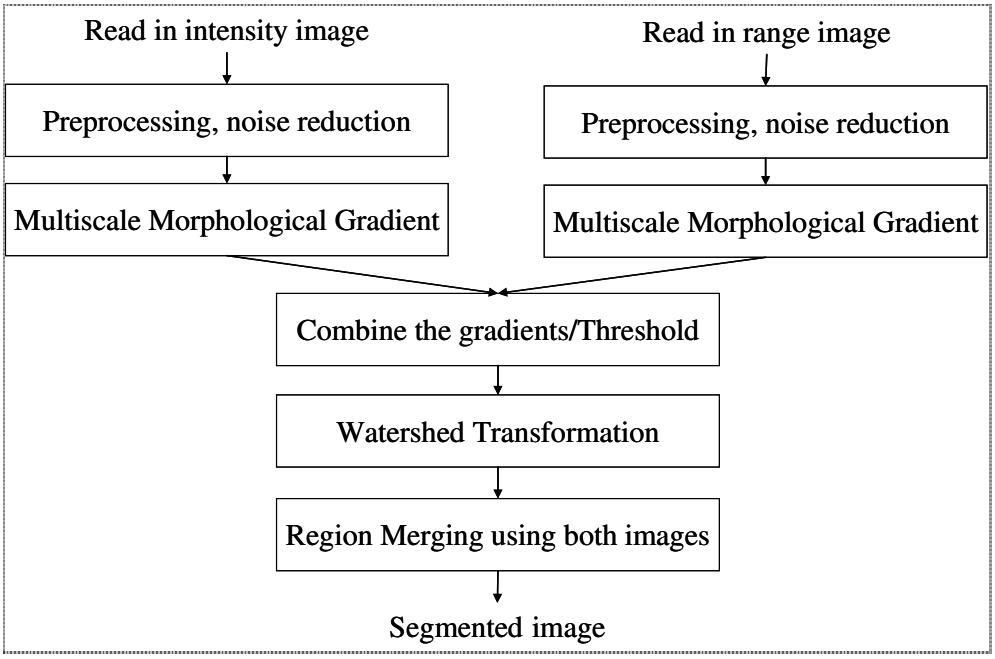


Figure 6.4: The Proposed System Diagram

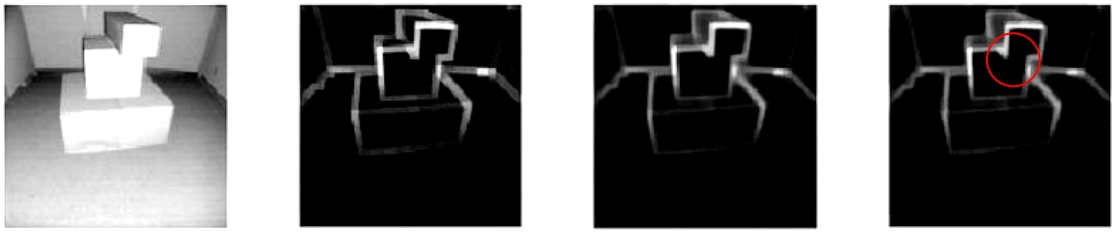
6.3 Simulation Results

The simulation is composed with two different cases to check the advantage of the proposed system. The first case uses only the intensity image to divide the given scene into homogeneous regions with proposed method. The second case utilizes both the intensity and range image to do the same. As the nature of image segmentation simulation, it is hard to analyze the statistical properties of the results. Consequently, we will evaluate the proposed systems by comparing the final result to human-eye ground truth.

6.3.1 Only-Intensity-Image (OIIM) case

In this case, only the intensity image is read in and processed with a given series of

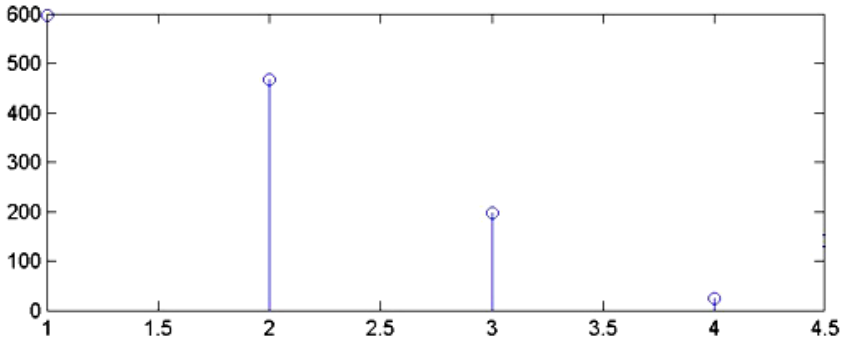
morphological processes using the left side of Figure 6.3 excluding combining edge information. Figure 6.5~7 show the results of each stage from the original read-in intensity image to the final segmented image. First of all, Figure 6.5 shows the effect of the multiscale morphological gradient to improve finding edge information from intensity image. As shown, the edges of the multiscale morphological gradient (c: Multiscale Morphological Gradient_MMG) is stronger and smoother than the non-multiscale gradient (b: Morphological Gradient_MG). The last one (d: Thresholded MMG_Th-MMG) shows the multiscale gradient after taking out trivial edges by thresholding. For the threshold, 30% of the value of 'Otsu's method' is used. Thresholded multiscale morphological gradient is actually fed into the watersheds transform function to do further processing for the proposed system.



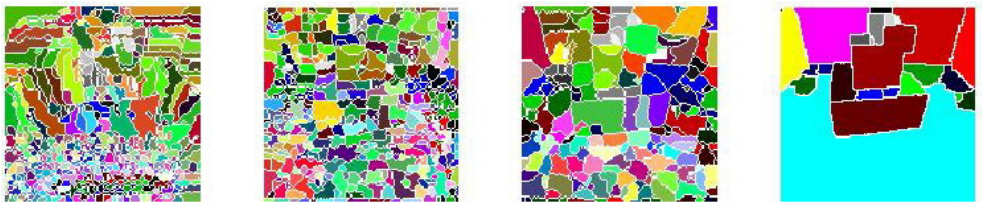
a)Original b)MG c)MMG d)Th-MMG

Figure 6.5: The Morphological Gradients

Applying the watersheds transformation on Th-MMG reduces the number of segments, which consequently affects the time it takes for region merging. The smaller the number of initial regions taken as input, the lower the computational expense cost, and the shorter the time it takes for the process. The bar graph of Figure 6.6 (a) shows the tendency that the number of segments is reduced as the multiscale and threshold processes are added in order. Part (b) shows the actual homogeneous regions of watersheds transformation.



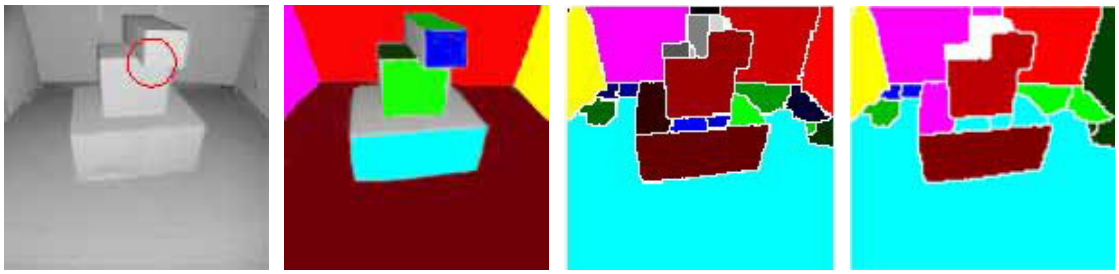
a) the tendency of the number of regions



b) the watersheds transformation for each image of Figure 6.5

Figure 6.6: Tendency of the Number of Segments after Watersheds Transformation

The given human-eye ground truth is shown in Figure 6.7 (b) and is composed of 11 different homogeneous regions. The initial homogeneous regions of the result of the watersheds transformation is set to be merged until the number of its regions match that of the ground truth in the post-processing stage. The final segmented image for the OIIM case with watersheds is shown in part (d).



a) Original b) Ground Truth c) before merging d) final

Figure 6.7: The Comparison between Ground Truth and Final Segmented Image (OIIM)

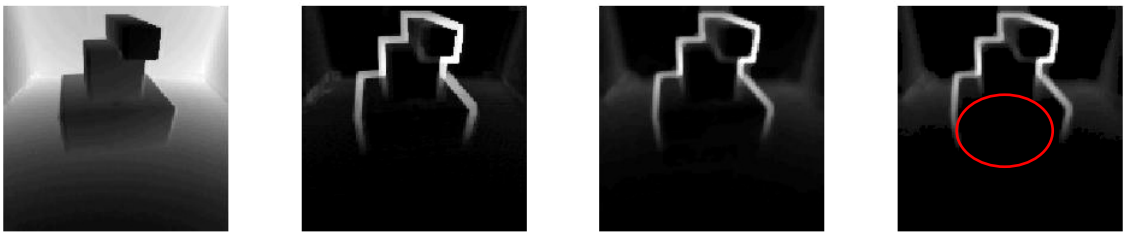
The final result does not perfectly match to the ground truth in OIIM case. It has some faults. One is that the sides and the face of the third box on the top are merged with the second box before merging (see the red circle in part (a) and the MMG of intensity image Figure 6.5). The intensity value of original image on that regions are very similar to each other; even though the multiscale morphological gradient is used, it cannot find the edges in that region (i.e., a hidden edge). So, the watersheds transformation considers those regions as a homogeneous region. Another fault is that the back wall and floor are merged with the first box. This is because the dissimilarity function of region merging is using only the average intensity value of each region. The last fault is that the shades at the corners are dominant for the intensity value on those regions irrelevant to original intensity values of the regions, even though the image is taken in a light-controlled environment.

The system shown in Figure 6.4 is proposed to improve these problems in the OIIM case like finding out hidden edges and compensating for the dissimilarity of the average intensity values.

6.3.2 Both-Intensity-Range-Image (BIRIM) case

For this case, the proposed system in Figure 6.4 is used fully. Both intensity and range images are read in, and the noise in these images is reduced using a 2-D Gaussian filter on each. The range image is linearly equalized with the maximum and minimum intensity values to match the contribution on the criteria with that of the intensity image. Later, both images are processed with the multiscale morphological gradient scheme separately, and two MMG are combined linearly with equal weight. The weight can be adjusted according to the situation. Then, the combined edge information is thresholded with same method that is used for the OIIM case. After watersheds transformation, the region merging stage uses both intensity and range images again to get the dissimilarity between each region.

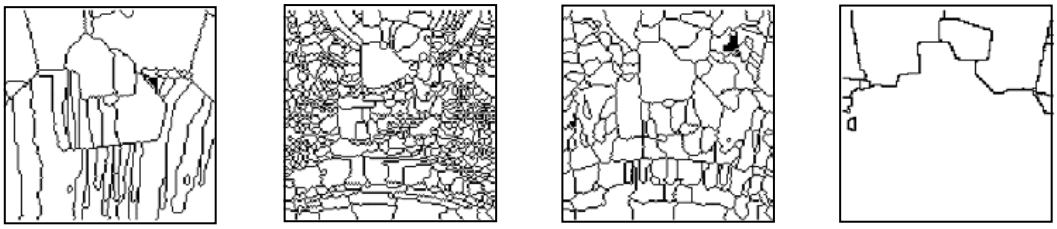
Figure 6.8 shows the original range image and its MG, MMG, and Th-MMG. It can be checked that the multiscale scheme also works on range data. The gradients of range image could have hidden edges (see the red circle in part (d)). Hopefully, it is expected not to overlap those of the intensity image. If it does, then it is not possible to reveal hidden edges in the sense of intensity and range.



a) Original b)MG c)MMG d)Th-MMG

Figure 6.8: The Gradients of Range Image

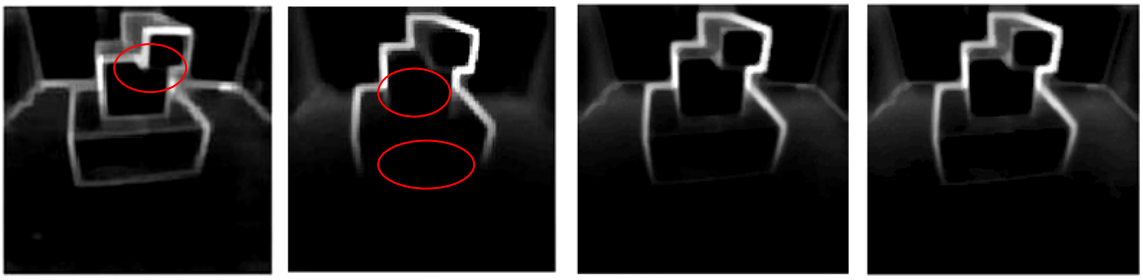
Figure 6.9 shows the initially segmented image applying the watersheds transformation on the MG, MMG and Th-MMG of the range image. Unlike the OIIM case, the number of segments after applying the transformation on the original range image is a small number. This is because the scene is composed of smooth planes like the floor, wall and boxes. Viewed in perspective angle, the smooth planes are changing their distance gradually along to the line of sight of camera and the elevation procedure in the watersheds transformation just fill in the region without building a dam in that direction. That is why it has vertically long segments and the edges do not contain the shape of the object. Even though there are some noises, they can be treated as 'islands' and it is erased through the watersheds transformation as reviewed in Chapter 3.



a) Original b) MG c) MMG d)Th-MMG

Figure 6.9: Watersheds Transformation on the Gradients of Range image

Next, the combination of both MMG-(CMMG) and threshold-(Th-CMMG) on the combination are shown in Figure 6.10. Hidden edges could be overlapped in intensity and range sense. Fortunately, we have non-overlap hidden edges in this example. The hidden edges in each intensity and range image are revealed in combined the MMG (CMMG)-part (c).



a) Int.-MMG b) Ran.-MMG c) CMMG d) Th-CMMG

Figure 6.10: MMG's and the CMMG and Th-CMMG

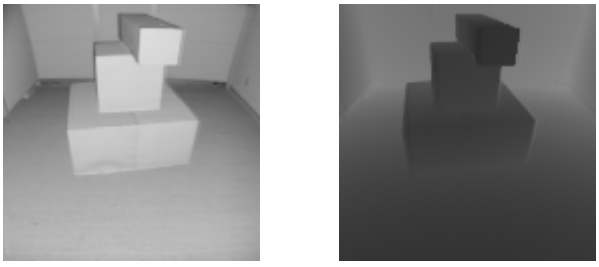
The watersheds transformation on the Th-CMMG is shown in Figure 6.11 compared to that of Th-MMG of the OIIM case. The number of regions of the watersheds transformation on Th-CMMG is slightly more than OIIM case, yet a fairly small number compared to the number of regions of watersheds transformation in original intensity image. It is natural that the number of regions of the watersheds transformation on Th-CMMG is

more than that of the OIIM case because Th-CMMG has more edges than Th-MMG if there were hidden edges.

a) OIIM case b) BIRIM case

Figure 6.11: Primitive Segments of OIIM and BIRIM Cases

The final result after the region merging process is shown in Figure 6.12 compared to the result of OIIM case and the ground truth. Three faces of the boxes are segmented properly in the BIRIM case; however, the top of first box is merged with the floor region and the dominant shadow is still affecting the segmentation at corners.



a) intensity image b) range image

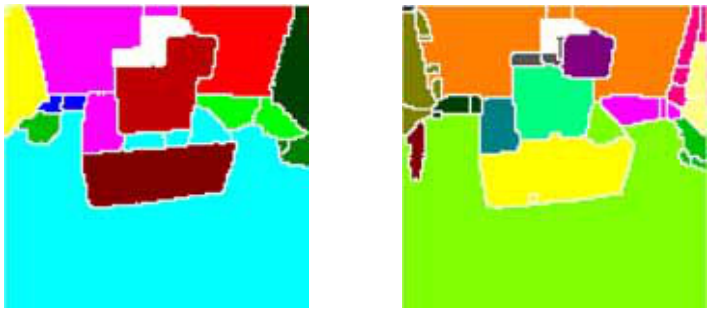
c) ground truth d) final in OIIM e) final in BIRIM

Figure 6.12: Comparison of Final Result with Ground Truth

6.3.3 Region-Merging-Stopping condition-Dissimilarity

In previous Sections 6.3.1 and 6.3.2, the number of segments of ground truth is used for the region-merging-stopping condition, so the region merging process is set to stop if the number of present segments is equal to that of the ground truth. However, this condition does not use any feedback from the present segment status at all, so it could merge two distinguishable regions forcefully. That is why we had objects merged with the floor or back wall. The process should be stopped before those situations occur. In this section, the dissimilarity value is used.

Figure 6.13 shows the results of the region merging process. Previous processes are applied in the same way as in the proposed system in Figure 6.4. The maximum dissimilarity value allowed to be merged is set to 12,000 for both OIIM and BIRIM case. The final numbers of regions for OIIM and BIRIM are 11 and 16.



a) OIIM case b) BIRIM case

Figure 6.13: The Results of Dissimilarity Case

As shown in 6.13, the back wall and the floor started to merge with the top of the first box in OIIM case, before the segments of back wall themselves are merged as a homogeneous region. On the other hand, in BIRIM case, the final segmenation shows good results except for the dominancy of the shadows in the intensity image.

CHAPTER 7

DISCUSSIONS AND FUTURE WORKS

Image segmentation is one of the most important categories of image processing. The purpose of image segmentation is to divide the original image into homogeneous regions. Image segmentation can be a pre-processing stage for other image processes and the result of the whole image processing depends on the result of the image segmentation. There are several approaches to perform this task such as Edge-based, Clustering-based, Region-based, and Split/merge approaches. In this work the watersheds transformation was selected as a particular region-based approach method to do the segmentation. The immersion process of the watersheds transformation is a fast and powerful algorithm to produce the segmented image. The transformation needs pre-processing and post-processing for embedded problems. The watersheds transformation may better applied on the gradient of the original image to get over the edge ambiguity. The morphological gradient and multiscale concepts are selected to find reliable edge information. Another embedded problem of the watersheds transformation is that it usually produces too many homogeneous regions. The number of segments can be reduced dramatically when the transformation is applied to the gradient image; however, this generally still results in a large number of segments. As a post-processing, the region merging algorithm is applied, using a region adjacency graph (RAG) and the dissimilarity function for the merging criteria.

In this thesis, the proposed system has been applied to co-aligned images, i.e., pair of intensity and range images. It is expected that the hidden edges in the sense of intensity can be detected in the sense of range or vice versa. Also it is expected that the

contribution of the range image in region merging can compensate for the dominancy of shadows in the intensity image irrelevant to original intensity of object. When hidden edges are not overlapped in both images, the proposed system detected those. This produced a slightly higher number of segments after the application of the watersheds transformation. The region merging algorithm worked somewhat, but the overall algorithm still needs to be improved. The calculation time for the RAG table and the dissimilarity between each region neighbor is very expensive; the RAG needs to be replaced with a Nearest Neighbor Graph [6] which is a faster and compact version of the RAG by finding the closest adjacent region out of its neighbor regions. Even though the range information is added to compensate the effect of shadows, the algorithm still merges wrong regions. The dissimilarity function used for the criterion of merging needs to be redefined including another statistical properties like variance. The dissimilarity can be defined in the sense of intensity, range and both separately, and the combination of these three dissimilarities can be used as criteria for region merging. This subject may be good a topic for future work.

Finally, there is another problem with the proposed system that should be pointed out. After combining two edge information from intensity and range images, the proposed system thresholds the combined gradient to eliminate local minima so that it can reduce the number of homogeneous regions of watersheds transformation. Valuable edge information could be lost through this process. To prevent the loss of this valuable edge information, morphological image reconstruction can be replaced with thresholding. Morphological image reconstruction is a powerful method to eliminate local minima while keeping all valuable edge information. The mathematical background is shown in Appendix B. This could be another subject for future work to incorporate morphological image reconstruction with the proposed system.

APPENDIX A

WATERSHEDS TRANSFORMATION ALGORITHM

The fast watersheds transformation algorithm was introduced by Luc Vincent and Pierre Soille in 1991. This pseudo-code is captured from the original paper [6]. This algorithm uses three different FIFO (first-in-first-out) data structures.

fifo_add(p): Puts the (pointer to) pixel *p* into the queue

fifo_first(): Returns the (pointer to) pixel which is at the beginning of the queue, and removes it from the queue

fifo_empty: Returns *true* if the queue is empty and *false* otherwise

A 'circular' queue is efficient in order to implement such operations. Below is some C-friendly pseudo code.

Algorithm: Fast Watersheds Transformation

```
#define MASK     -2    /*initial value of a threshold level*/
#define WSHED    0     /*value of the pixels belonging to the watersheds*/
#define INIT     -1        /* initial value of im_o*/
```

• ---input: im_i, decimal image;
  ---output: im_o, image of the labeled watersheds;
• Initializations:

---Value INIT is assigned to each pixel of im_o: $\forall p \in D_{im_o}, im_o(p) = INIT$;

---*current_label* ← 0;

---*current_dist*: integer variable;

---im_d: work image (of distances), initialized to 0;

• Sort the pixels of *imi* in the increasing order of their gray values.

Let h_{min} and h_{max} designate the lowest and highest values, repectively.

• For $h \leftarrow h_{min}$ to h_{max} {

/* geodesic SKIZ of level *h-1* inside level *h* */

For every pixel *p* such that $im_i(p) = h$ {

/* These pixels are accessed directly through the sorted array. */

$im_o(p) \leftarrow$ MSAK;

if there exists $p' \in N_G(p)$ *such that* $im_o(p') > 0$ *or* $im_o(p') = WSHED$ {

$im_d(p) \leftarrow 1$; *fifo_add(p)*;

}

}

current_dist ← 1; *fifo_add(fictitious_pixel)*;

repeat indefinitely {

p ← *fifo_first()*;

if *p* = *fictitious_pixel* {

if *fifo_empty() = true* then BREAK;

else { *fifo_add(fictitious_pixel)*;

current_dist ← *current_dist* + 1;

p ← *fifo_first()*;

}

}

For every pixel *p'* ∈ *NG(p)* {

If $im_d(p') <$ *current_dist* and ($im_o(p') > 0$ or $im_o(p')$ = WSHED){

```
            /*i.e., p' belongs to an already labeled basin or to the watersheds*/
            if imo(p') > 0 {
                if im_o(p) = MASK or im_o(p) = WSHED then
                    im_o(p) ← im_o(p');
                else if im_o(p) ≠ im_o(p') then
                    im_o(p) ← WSHED;
            }
            else if im_o(p) = MASK then im_o(p) ← WSHED
            }
        else if im_o(p') = MASK and im_d(p') = 0 {
            im_d(p') ← current_dist +1; fifo_add(p');
        }
    }
}
/* checks if new minima have been discovered*/

For every pixel p such that im_i(p) = h {
    im_d(p) ← 0;        /* the distance associated with p is reset to 0*/
    if im_o(p) = MASK {
        current_label ← current_label +1;
        fifo_add(p); im_o(p) ← current_label;
        while fifo_empty() = false{
            p' ← fifo_first();
            For every pixel  p" ∈ NG(p') {
                if im_o(p") = MASK { fifo_add(p");
                    im_o(p") ← current_label; }
            }
```

```
                }
            }
        }
    }
```

APPENDIX B
LOCAL MINIMA ELIMINATION: MORPHOLOGICAL IMAGE RECONSTRUCTION

Local minima consist of a small number of pixels or have a low contrast with respect to their neighbors. It is usually caused by noise or quantization error. The procedure to eliminate local minima used in [4] takes advantage of the technique "morphological grayscale reconstruction" proposed in [5].

The morphological reconstruction transformation is well-known in the binary case, where it simply extracts the connected components of an image which are "marked" by another image. Extending it to grayscale reconstruction, it can accomplish several tasks such as image filtering, extrema, domes, and basins extraction. In the following paragraphs, we will review morphological reconstruction first and then describe how to apply it to modify the gradient.

Definition: Binary Reconstruction

Let $X,Y \subset Z^2$ and $Y \subseteq X$. The reconstruction of X from Y is obtained by iterating elementary geodesic dilations of Y inside X until stability is reached; that is,

$$\rho_X(Y) = \bigcup_{n \geq 1} \delta_X^{(n)}(Y).$$

where $\delta_X^{(n)}(Y)$ can be obtained by iterating n elementary geodesic dilation and the geodesic dilation is defined as:

$$\delta_X^{(1)}(Y) = (Y \oplus B) \bigcap X \ (B \text{ is the structuring element of size } 1).$$

Definition: Dilation-based Grayscale Reconstruction

Let I, J be two grayscale images defined on the same domain and $J \leq I$. The reconstruction of I from J, denoted as $\gamma_I^{(rec)}(J)$, is obtained by iterating elementary geodesic dilations of J under I until stability is reached:

$$\gamma_I^{(rec)}(J) = \underset{n \geq 1}{V} \delta_I^{(n)}(J)$$

where $\delta_I^{(n)}(J)$ can be obtained by iterating n elementary geodesic dilation and the geodesic dilation is defined as

$$\delta_I^{(1)}(J) = (J \oplus B) \Lambda I$$

(B is the flat structuring element of size 1 and Λ stands for pointwise minimum.)

Definition: Erosion-based Grayscale Reconstruction

Let I, J be two grayscale images defined on the same domain and $J \leq I$. The reconstruction of I from J, denoted as $\varphi_I^{(rec)}(J)$, is obtained by iterating elementary geodesic erosions of J above I until stability is reached :

$$\varphi_I^{(rec)}(J) = \underset{n \geq 1}{\Lambda} \varepsilon_I^{(n)}(J)$$

where $\varepsilon_I^{(n)}(J)$ can be obtained by iterating n elementary geodesic dilation and the geodesic dilation is defined as :

$$\varepsilon_I^{(1)}(J) = (J \Theta B) V I$$

(B is the flat structuring element of size 1 and V stands for pointwise maximum)

Now is the time to describe how the morphology reconstruction helps to eliminate the local minima. Generally, grayscale dilation has the effect of eliminating specks in a signal, and it can be employed to smooth the gradient image. Given the gradient image by applying the multiscale gradient algorithm in the last section, the prodecure of local minima elimination in [4] can be stated as follows:

1. The gradient image *MG(f)* is dilated with a square structuring element B_s of 2x2 pixels, i.e., $MG(f) \oplus B_s$.
2. A constant h is added to the dilated gradient image and erosion-based reconstruction is performed. That is, the final gradient image can be expressed as $\varphi^{(rec)}_{(MG(f))}((MG(f) \oplus B_s) + h)$.

As an interpretation, in step 1, if the local minimum is 'narrower' than the size of B_s, it will be filled by the nature of dilation. In step 2, the local minimum with intensity in gradient lower than h can be filled irrespective of their absolute value. If the local minimum is wide and deep, it still cannot be removed. However, compared to thresholding the gradient image, it is more reasonable to use the morphological approach shown above to eliminate the local minima.

BIBLIOGRAPHY

1. Philippe Salembier, "Morphological multiscale segmentation for image coding," *Signal Processing*, 38:359-386, 1994.

2. Malay K. Kundu, Bhabatosh Chanda and Y. Vani Padmaja. "A multiscale morphologic edge detector," *Pattern Recognition*, 31(10):1469-1478, 1998.

3. S.J. Lee, R.M Haralick and L. G. Shapiro, "Morphologic edge detection," *IEEE J. Robot. Automat.*, 3(2): 142-155, 1987.

4. Demin Wang, "A multiscale gradient algorithm for image segmentation using watersheds," *Pattern Recognition*, 30(12):2043-2052, 1997

5. Luc Vincent, "Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms," *IEEE Trans. on Image Processing*, 2(2):176-201, 1993.

6. Luc Vincent and Pierre Soille, "Watersheds in digital space: An efficient algorithm based on immersion simulations," *IEEE Tran. on Pattern Recognition and Machine Intelligence*, Vol. 13, No. 6, June 1991.

7. H. Digabel and C. Lantuéjoul, "Iterative algorithms," *Proc. 2nd European Symp. Quantitive Analysis of Microstructures in Material Science, Biology and Medicine*, October 1977.

8. S. Beucher and C. Lantuéjoul, "Use of watersheds in contour detection," *Proc. Int. Workshop Image Processing, Real-time Edge and Motion Detection/Estimation*, September 1979.

9. S. Beucher, “Watersheds of functions and picture segmentation,” *Proc. IEEE. Int. Conf. Acoustics, Speech, and Signal Processing*, May 1982.

10. Charles R. Giardina and Edward R. Dougherty, Morphological Methods in Image and Signal Processing, Prentice-Hall, Inc., 1988.

11. Petros Maragos, “Tutorial on advances in morphological image processing and analysis,” *Optical Engineering*, 26(7):623-632, July 1987.

12. J. Serra, Image Analysis and Mathematical Morphology, Academic Press Inc., 1982.

13. Rafael C. Gonzalez, Richard E. Woods, Digital Image Processing 2nd, Prentice-Hall Inc, 2002.

14. J. Beaulieu and M. Goldberg, “Hierarchy in picture segmentation: A stepwise optimization approach,” *IEEE Trans. Pattern Anal. Machine Intel.*, 11:150-163, Feb. 1989.

15. J. H. Ward, “Hierarchical grouping to optimize an objective function,” *J. American Stat. Assoc.*, 58:236-245, 1963.

16. K. Haris, “A hybrid algorithm for the segmentation of 2D and 3D images,” Master’s thesis, University of Crete, 1994.

17. Serge Beucher, “The watershed transformation page: Image segmentation and mathematical morphology,” cmm.ensmp.fr/~beucher/wtshed.html.

BIOGRAPHICAL SKETCH

Hyun Geun Yu was born on April 11, 1975 in Dain, South Korea. After graduating from Sogang University in 2001 with a Bachelor's degree in Electrical Engineering, he came to United States America and continued his study and research at Florida State University. He worked on projects funded by National Imagery and Mapping Agency (NIMA) and the US Army Robotics Collaborative Technology Alliance (CTA) for about 3 years and received his Master of Science degree in the Fall of 2004. He will continue his Ph.D program in the area of robotics at the same school.