AUTOMATIC MATCHING OF AERIAL COASTLINE IMAGES WITH MAP DATA

A Thesis Submitted to
the Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in Electrical and Electronics Engineering

by
Metin KAHRAMAN

October 2005
İZMİR

We approve the thesis of **Metin KAHRAMAN**

Date of Signature

.....
Asst.Prof. Şevket GÜMÜŞTEKİN
Supervisor
Department of Electrical and Electronics Engineering
İzmir Institute of Technology

20 October 2005

.....
Prof. F.Acar SAVACI
Department of Electrical and Electronics Engineering
İzmir Institute of Technology

20 October 2005

.....
Asst.Prof. Mustafa A. ALTINKAYA
Department of Electrical and Electronics Engineering
İzmir Institute of Technology

20 October 2005

.....
Asst.Prof. Serdar ÖZEN
Department of Electrical and Electronics Engineering
İzmir Institute of Technology

20 October 2005

.....
Asst.Prof. Bedrettin SUBAŞILAR
Department of Physics
İzmir Institute of Technology

20 October 2005

.....
Prof. F.Acar SAVACI
Head of Deparment
Department of Electrical and Electronics Engineering
İzmir Institute of Technology

.....
Assoc.Prof. Semahat ÖZDEMİR
Head of the Graduate School

ACKNOWLEDGEMENTS

I would like to express my sincere thanks to my supervisor Asst.Prof. Şevket Gümüştekin for his invaluable support, guidance and patience in the preparation of this thesis. I would like to acknowledge with gratitude the technical and editorial guidance given the in revision of this thesis by Prof. Ferit Acar Savacı, Asst.Prof. Mustafa Aziz Altınkaya, Asst.Prof. Serdar Özen and Asst.Prof. Bedrettin Subaşılar.

Many thanks go to Lieutenant-Commander Haldun Karatepe who provided the vector map data.

I am grateful for Lieutenant-Junior Grade Cafer Karabulut and Lieutenant-Junior Grade Bora Bilgiç who compensated for my absence at work. I am deeply indebted to my commander Captain Atilla Oktay for his support.

I am very grateful for my parents for their continuous support.

Last but not least, I would like to give special thanks to my wife Selda. Without her support, patience and love I could never have completed this thesis.

ABSTRACT

Matching aerial images with map data is an important task in remote sensing applications such as georeferencing, cartography and autonomous navigation of aerial vehicles. The most distinctive image features that can be used to accomplish this task are due to the unique structures of different coastline segments. In recent years several studies are conducted for detecting coastlines and matching them to map data. The results reported by these studies are far from being a complete solution, having weak points such as poor noise sensitivity, need for user interaction, dependence to a fixed scale and orientation.

In this thesis, a two-step procedure involving automatic multiresolution coastline extraction and coastline matching using dynamic programming have been proposed. In the proposed coastline extraction method, sea and land textures are segmented by using cooccurrence and histogram features of the wavelet image representation. The coastlines are identified as the boundaries of the sea regions. For the coastline matching, shape descriptors are investigated and a shape matching method using dynamic programming is adapted. Proposed automatic coastline extraction and coastline matching methods are tested using a vector map of the Aegean coast of Turkey.

ÖZET

Havadan alınan görüntülerinin harita verileri ile çakıştırılması haritacılık ve hava araçlarının insansız seyri gibi uzaktan algılama uygulamalarında önemli bir yer alır. Bu işin gerçekleştirilebilmesi için gerekli olan en ayırt edici görüntü öznitelikleri, farklı kıyı bölütlerinin özgün yapılarıdır. Son yıllarda, kıyı görüntülerinden kıyıları belirleme ve harita verileri ile çakıştırma konusunda çeşitli araştırmalar yapılmıştır. Bu araştırmalardan çıkarılan sonuçların, gürültü hassasiyeti, kullanıcı gerekliliği, değişmez ölçek ve yönelime bağlılığı gibi zayıf tarafları nedeniyle şu ana kadar tam bir çözüme ulaşılmadığı görülebilir.

Bu tezde, otomatik çoklu çözünürlüklü kıyı çıkarma ve dinamik programlamayı kullanan kıyı çakıştırma yöntemlerini içeren iki basamaklı bir yordam önerilmektedir. Önerilen kıyı çıkarma yönteminde, deniz ve kara dokuları dalgacık görüntü gösteriminin birliktelik matrisi ve histogram öznitelikleri kullanılarak bölütlenmektedir. Kıyılar, deniz bölgelerinin sınırları olarak belirlenir. Kıyı çakıştırma için, şekil tanımlayıcılar araştırılmış ve dinamik programlamayı kullanan bir şekil çakıştırma yöntemi uyarlanmıştır. Önerilen otomatik kıyı belirleme ve kıyı çakıştırma yöntemleri Türkiye'nin Ege kıyısının vektör haritası üzerinde denenmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Table **Page**

LIST OF ABBREVIATIONS

C	Convex
CAFD	Cumulative Angular Function Fourier Descriptor
CBIR	Content-Based Image Retrieval
CCFD	Complex Coordinates Fourier Descriptor
CCH	Chain Code Histogram
CDFD	Centroid Distance Fourier Descriptor
CFD	Curvature Fourier Descriptor
CSS	Curvature Scale Space
CSSD	Curvature Scale Space Descriptor
CM	Cooccurrence Matrix
CWT	Continuous Wavelet Transform
DC	Direct Current (constant term)
DFT	Discrete Fourier Transform
DP	Dynamic Programming
DWT	Discrete Wavelet Transform
FD	Fourier Descriptor
FLDA	Fisher's Linear Discriminant Analysis
FIR	Finite Impulse Response
GPS	Global Positioning System
ML	Maximum Likelihood
MPEG	Moving Picture Experts Group
RIV	Relative Importance Value
UAV	Unmanned Air Vehicle
V	Concave

CHAPTER 1

INTRODUCTION

Coastline extraction and coastline matching with map data are important issues in several applications and coastal studies such as autonomous navigation of aerial vehicles, cartography, georeferencing satellite images and coastal geomorphology monitoring. Although there are previous studies on coastline extraction and matching the techniques developed by these studies are known to be non-automatic, time-consuming and not suitable for autonomous navigation. In (Bo et al. 2000) and (Bo et al. 2001) fuzzy rules and textural features are used in a semiautomatic scheme to extract coastlines. In (Zhang et al. 1997), (Loos et al. 2002) and (Jishuang et al. 2002) edge based extraction is done in high contrast aerial images. (Jianbin et al. 2003) uses Hausdorff distance to match coastal features found by edge detection. (Eugenio et al. 2002) employs a contour based matching to register aerial images. (Bijaoui and Cauneau. 1994) uses spectral and textural features to extract coastlines in SAR images.

Using an automatic coastline matching method as a part of contour matching system can be useful for the autonomous navigation of an unmanned air vehicle (UAV) when no GPS data and no communication are available. A passive navigation system is also very important in military UAV applications

In cartography, updating coastline maps and charts by using recent aerial coastline images requires extracting and matching coastlines in the images with old map data. Cataloging satellite images and monitoring coastal geomorphology, both use multitemporal, multisatellite and multisensor coastline images. In order to index and analyze these images, the images should be georeferenced by automatically matching images with map data.

In this thesis, problem of matching of aerial coastline images with map data has been studied because of the needs described above. The problem has been studied in two main parts: Automatic multiresolution coastline extraction and matching of extracted coastlines with map data.

For proposed automatic coastline extraction method, first we implemented the wavelet decomposition of the coastline image to extract multiresolution information and

to highlight textural content. The wavelet image has been segmented into sea and land regions by using texture segmentation. Multiscale segmentation of the wavelet image has been achieved by means of coarse to fine segmentation which employs coocurrence features and histogram features of textures. Cooccurrence features have been selected by using a feature selection scheme based on Fisher's linear discriminant analysis (FLDA). Sea and land textures have been classified by using the maximum likelihood classifier. After classification of sea region, boundary of sea region has been extracted as the coastline.

In the matching of extracted coastlines with map data, shape descriptors have been studied to select a matching algorithm which is suitable for coastline matching. Some contour-based shape descriptors such as chain code representation, Fourier descriptors, and curvature scale space descriptor have been analyzed in the aspects of invariance to image transformations and convenience to open curve matching.

Proposed coastline matching method in this thesis has been adapted from Petrakis, Diplaros and Millios' shape retrieval algorithm (Petrakis et al. 2002). In the proposed method, extracted coastlines are segmented into convex and concave segments by finding inflection points of the coastlines. Segment features called as turning angle, length, and area of the segments are extracted from the coastline segments. By use of Dynamic Programming table, extracted coastlines are matched with map data. Automatic coastline extraction and matching methods are tested on a vector map of Aegean coast of Turkey.

Main contributions of this thesis can be given as follows

1. Discriminating power of statistical texture features such as coocurrence and histogram features have been tested for sea and land textures by using the wavelet representation of the image.
2. Most promising window resolutions for coocurrence and histogram features have been found.
3. The performance of the maximum likelihood classifier has been tested on natural textures.
4. An automatic coastline extraction method has been developed.
5. Shape descriptors have been investigated and discussed their matching performance due to the open curve matching.
6. A shape matching method has been adapted for coastline matching

The first chapter of this thesis introduces motivation, thesis scope, and main

contributions. Chapter 2 presents the proposed automatic multiresolution coastline extraction method. In this chapter, the wavelet representation, statistical texture features, feature selection scheme, Maximum likelihood classifier have been described. Chapter 3 discusses shape descriptors. Chapter 4 presents the proposed coastline matching algorithm using dynamic programming and includes matching results. Chapter 5 is the conclusion of the thesis.

CHAPTER 2

AUTOMATIC COASTLINE EXTRACTION FROM AERIAL IMAGES

Accurate coastline information is necessary for updating coastal maps from recent remote sensing images, for detecting changes in coastlines in coastal geomorphology and for matching coastlines to map data for autonomous navigation.

In this thesis we have developed an automatic coastline extraction method to acquire coastline information from aerial coastline images. The acquired data is used for automatic matching to map data.

Since coastlines are basically the boundary of sea, we can consider coastline extraction problem as object boundary detection problem. So some methods, such as edge detection and texture analysis, which are usually used to extract object boundaries, can be employed in coastline extraction. Edge detection methods require user interaction for defining background and they find spurious edges. Since detection of sea texture in the image is sufficient for coastline extraction, texture analysis seems more convenient for automatic extraction of coastline extraction.

Aerial coastline images which may be taken from different altitudes contain different kinds of sea textures. Thus textures should be analyzed at different scales by using multiscale representation of texture.

In this thesis, we utilize statistical wavelet texture features which are coocurrence features, wavelet histogram features (Mallat 1989), energy features in order to classify sea textures in the images.

By using coarse to fine detection of sea texture, an image is inspected starting with large blocks which look for second order statistics (coocurrence features) of sea texture and then smaller blocks that look for first order statistics (histogram features). Finally very small blocks which look for energy signatures are used.

Examining of the sea texture by large blocks involves Haralick's textural features (Haralick et al. 1973) which are extracted from coocurence statistics of wavelet coefficients of multiscale wavelet image. Since higher dimensionality of feature causes higher computational complexity and classification performance does not increase with the increasing number of feature set, an appropriate feature selection method is

employed in order to select an optimal feature set for a particular classification problem. In this thesis, Fisher's Linear Discriminant Analysis (FLDA) is adopted to obtain an optimal feature set from all textural features.

As the second step of texture inspection, smaller blocks are used. Haralick's textural features are observed to give misleading results to small window size. In this step histogram signatures that found experimentally by Mallat (Mallat 1989) for modeling the detail histograms of natural textured wavelet images by a family of histograms, are employed as the texture features.

Since texture is characterized by high-frequency components of the image, only detail images of wavelet image are used for examining sea texture.

2.1. The Wavelet Representation

In image processing, it is difficult to analyze the content of an image directly from gray-level intensity of the pixels in the image. Gray values of the pixels depend on the lighting conditions. Local variations of the image intensity are more important. The size of window in which we analyze the image determines the size of the structures that we want to analyze. The size of the window defines a resolution for computing the local variations of the image. The scale of the image varies with the distance between the scene and the sensor. Thus the structures we want to detect have different sizes. For this reason, the image information should be represented at different resolutions. This representation is called the multiresolution representation of the image.

The wavelet representation is a powerful multiresolution decomposition method that represents details with different spatial orientations (such as edges with horizontal, vertical, and diagonal orientations). In image processing, the wavelet representation of an image is obtained by applying two-dimensional wavelet transform to the image. In this section, the wavelet transform is first described for one-dimensional continuous and discrete signals and then extended to two-dimensions for the images.

2.1.1. The Continuous Wavelet Transform (CWT)

In Fourier theory, a signal can be expressed as the sum of an infinite series of sines and cosines. This sum is also called to as a Fourier transform. The big disadvantage of a Fourier transform is that it has only frequency resolution and no time resolution. In order to overcome this problem, in wavelet analysis the use of a scalable modulated window is shifted along the signal and for every position the spectrum is computed. This process is repeated with a different size of windows for every resolution. At the end of the process, a set of time-frequency representations of the signal at different resolutions is obtained as a wavelet transform of a signal. For a continuous signal f(t), the continuous wavelet transform (CWT) of f(t) can be formally represented by:

$$W(s,\tau) = \int f(t)\psi^{*}_{s,\tau}(t)dt \qquad (2.1)$$

where " * " denotes complex conjugation and the variables s and τ are scale and translation factors, respectively . In (2.1), f(t) is decomposed into a set of basis functions $\psi_{s,\tau}(t)$, called the wavelets. The wavelets are generated from a single basic wavelet $\psi(t)$, called mother wavelet, by scaling and translation:

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-\tau}{s}\right). \qquad (2.2)$$

2.1.2. The Discrete Wavelet Transform (DWT)

The main idea of the discrete wavelet transform is the same as that of the CWT. A time-frequency representation of a discrete signal is obtained using discrete filtering techniques. The continuous wavelet transform is computed by changing the scale of the

window, translating the window in time, multiplying by the signal, and integrating over all times. In the discrete case, filters of different cutoff frequencies are used to analyze the signal at different scales. The signal is passed through a series of high pass filters to analyze the high frequencies, and it is passed through a series of low pass filters to analyze the low frequencies. The resolution of the signal, which is a measure of the amount of detail information in the signal, is changed by the filtering operations, and the scale is changed by upsampling and downsampling. The DWT analyzes the signal at different frequency bands with different resolutions by decomposing the signal into a coarse approximation and detail information. The DWT employs two sets of functions, called scaling functions, "φ", and wavelet functions, "ψ", which are associated with lowpass and highpass filters, respectively (Mallat 1989):

$$\varphi_{j,k}(t) = \varphi(2^j t - k) = \sum_n h(n-2k)\sqrt{2}\,\varphi(2^{j+1}t - n) \tag{2.3}$$

$$\psi_{j,k}(t) = \psi(2^j t - k) = \sum_n g(n-2k)\sqrt{2}\,\varphi(2^{j+1}t - n) \tag{2.4}$$

where j and k denote scale and translation, respectively. Scaling function represents coarse approximation of the signal while wavelet function represents the detail information of the signal. The signal, f(t), can be expressed in terms of scaling and wavelet functions :

$$f(t) = \sum_k c_{j_0}(k)\,2^{j_0/2}\varphi(2^{j_0}t - k) + \sum_k \sum_{j=j_0}^{\infty} d_{j-1}(k)\,2^j \psi(2^j t - k) \tag{2.5}$$

where $c_j(k)$ and $d_j(k)$ are the scaling and wavelet coefficients, respectively. If the scaling functions $\varphi_{j,k}(t)$ and the wavelets $\psi_{j,k}(t)$ are orthonormal, then the coefficients $c_j(k)$ and $d_j(k)$ are computed by ,

$$c_j(k) = \sum_n h(n-2k)c_{j+1}(n) \tag{2.6}$$

$$d_j(k) = \sum_n g(n-2k)c_{j+1}(n). \tag{2.7}$$

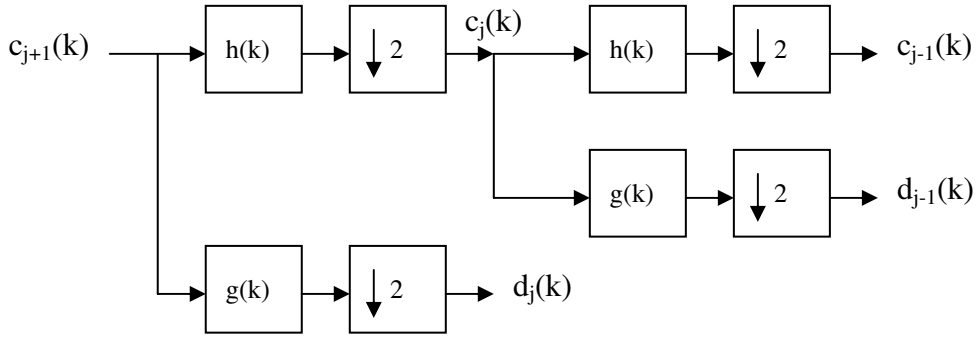


Figure 2.1. Filterbank implementation of two-level DWT.

2.1.3. Two-Dimensional Discrete Wavelet Transform

The concepts for the wavelet representation of one-dimensional signals can be easily extended to two dimensional wavelet representations of the images. The associated two-dimensional scaling function can be expressed as a separable function by $\varphi(x,y) = \varphi(x)\varphi(y)$ where $\varphi(x)$ and $\varphi(y)$ is the one-dimensional scaling functions (Mallat 1989). Then, three two-dimensional wavelet functions can be expressed as:

$$\psi^1(x,y) = \varphi(x)\psi(y) \quad \psi^2(x,y) = \psi(x)\varphi(y) \quad \psi^3(x,y) = \psi(x)\psi(y). \tag{2.8}$$

Two-dimensional DWT can be implemented by using the FIR filters similar to the one-dimensional DWT described in previous section. The implementation of two-dimensional DWT for one level is shown in Figure 2.2. The rows of the image, I(x,y), are first convolved with one-dimensional filters which H and G are low and bandpass quadrature mirror FIR filters, respectively, then the columns of the resulting images are

convolved with one-dimensional filters, H and G, again. Thus we obtain one-level two-dimensional DWT of the image, I(x,y), as one coarse image, L(x,y), and three detail images, $D_1(x,y)$, $D_2(x,y)$, $D_3(x,y)$.

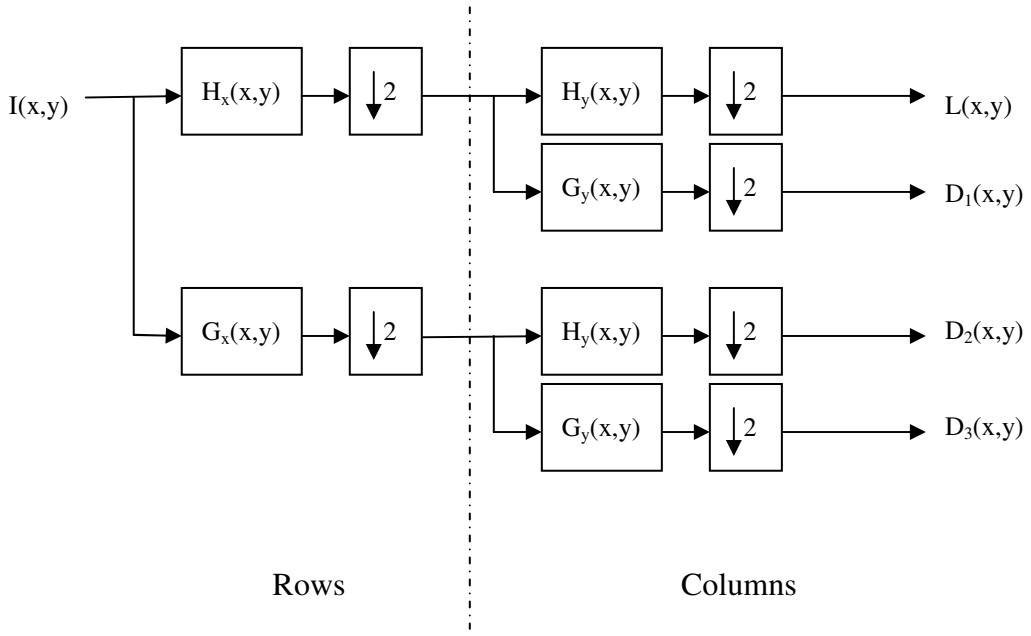


Figure 2.2. One-level two-dimensional discrete wavelet decomposition.

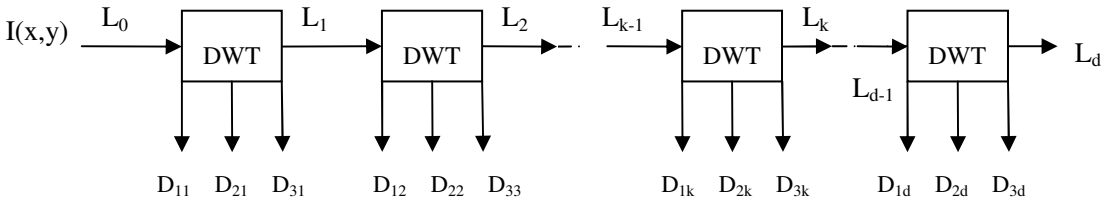


Figure.2.3. Multiresolution wavelet representation of the image k=1,2,…..,d.

In order to obtain multiresolution wavelet representation of the image, one-level two-dimensional DWT procedure is repeated n times by taking the coarse image, L(x,y), of the previous level as the input of the procedure where n is the number of the levels of the wavelet representation. This process is described in Figure.2.3. Multiresolution wavelet representation of the image can be formally described as:

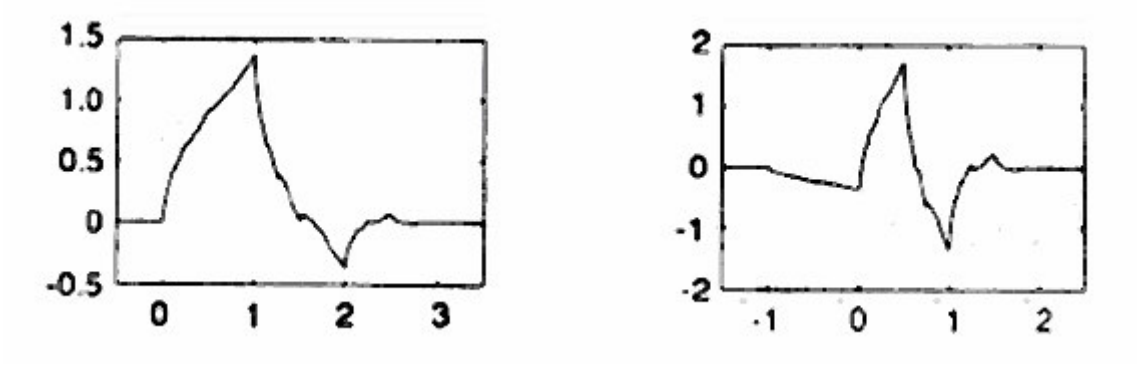
$$L_k(x,y) = \left[H_x * [H_y * L_{k-1}(x,y)]_{|2,1} \right]_{|1,2} \tag{2.9}$$

$$D_{k1}(x,y) = \left[H_x * [G_y * L_{k-1}(x,y)]_{|2,1} \right]_{|1,2} \tag{2.10}$$

$$D_{k2}(x,y) = \left[G_x * [H_y * L_{k-1}(x,y)]_{|2,1} \right]_{|1,2} \tag{2.11}$$

$$D_{k3}(x,y) = \left[G_x * [G_y * L_{k-1}(x,y)]_{|2,1} \right]_{|1,2} \tag{2.12}$$

where " * " denotes the convolution operator and "$|a,b$" is the decimation by (a,b) along (rows,columns); H and G are low and bandpass quadrature mirror FIR filters, respectively. L_0 is the original image and L_k is the low resolution image at scale k. The detail images D_{ki} (i=1,2,3) have detail information of the original image in horizontal, vertical and combined directions, respectively at scale k. The wavelet decomposition of the original image at d scales is called a multiscale representation of the original image at depth d.



(a) (b)

Figure 2.4. Daubechies 4-tap (a) scaling and (b) mother wavelet functions

In this work, an overcomplete wavelet representation has been used to avoid losing texture information at higher scales. This increases the accuracy of texture segmentation. Also, the wavelet transform is not translation invariant. The lack of translation invariance and the loss of the texture information can be avoided by omitting the decimations at the output of the filterbanks. The well-known wavelet filter of four-

tap Daubechies (Daubechies 1992), whose scaling and mother wavelet functions are shown in Figure 2.4, has been used as quadrature mirror FIR filters since these filters are used widely because of their orthogonal properties. The coefficients of the filter are shown in Table 2.1. High pass filters are modulated versions of low pass filters so such filters are called as quadrature mirror filters.

$$G(n)=(-1)^{-n} H(k-n) \qquad \text{where k is an integer delay.} \tag{2.13}$$

Table 2.1. Daubechies 4-tap filter coefficients.

Tap	Low Pass Filter	High Pass Filter
0	0.4830	0.1294
1	0.8365	0.2241
2	0.2241	-0.8365
3	-0.1294	0.4829

2.2. Feature Extraction for Texture Segmentation

Texture analysis plays an important role in image processing and remote sensing tasks. Correct classification of textural segments is highly dependent of a good set of features. In recent years, several methods have been proposed for texture feature extraction. A major group of texture feature extraction methods depends on the assumption that texture can be defined by local statistical features of pixel values. In most cases features are derived from first order (histogram) and higher order (cooccurrence) statistics.

2.2.1. Cooccurrence Features

Homogeneous regions can be effectively represented by first order statistics.

However in practice a spatial arrangement of pixels require the use of higher order statistics. Cooccurrence features efficiently reflect second order statistics of texture. They describe periodic structure of the texture when the size of texture increases. Cooccurrence features are computed from cooccurrence matrix to define texture properties.

2.2.1.1. Cooccurrence Matrix and Haralick's Textural Features

Cooccurrence matrix (CM), first introduced by Haralick (Haralick et al.73), is an effective representation for the second order statistics of textures. CM is a square matrix whose elements represent the occurrence frequency of values (gray levels) of pixel pairs separated by a fixed distance in a fixed orientation in a sample of texture. At 0^0, 45^0, 90^0, 135^0 directions and at d pixel distance CM can be represented by:

$$CM^{d}_{0^0}(a,b) = \#\left\{\left[(k,l),(m,n)\right] \middle| (k-m=0, |l-n|=d), f(k,l)=a, f(m,n)=b\right\} \tag{2.14}$$

$$CM^{d}_{45^0}(a,b) = \#\{\left[(k,l),(m,n)\right] \middle| (k-m=d, l-n=-d) OR\ (k-m=-d, I-n=d), \\ f(k,l)=a, f(m,n)=b\} \tag{2.15}$$

$$CM^{d}_{90^0}(a,b) = \#\{\left[(k,l),(m,n)\right] \middle| (|k-m|=d, l-n=0), f(k,l)=a, f(m,n)=b\} \tag{2.16}$$

$$CM^{d}_{135^0}(a,b) = \#\{\left[(k,l),(m,n)\right] \middle| (k-m=d, l-n=d) OR (k-m=-d, I-n=-d), \\ f(k,l)=a, f(m,n)=b\} \tag{2.17}$$

0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3

(a)

$$CM^{1}_{0^0} = \begin{bmatrix} 4 & 2 & 1 & 0 \\ 2 & 4 & 0 & 0 \\ 1 & 0 & 6 & 1 \\ 0 & 0 & 1 & 2 \end{bmatrix}$$

(b)

Figure 2.5. Cooccurrence Matrix (a) Gray level image and (b) its coocurrence matrix at one pixel distance and at 0^0 orientation.

where (k,l) and (m,n) are pixel locations, which are separated by a distance d and whose values are a and b, respectively.

In Figure 2.5, as an example, CM of a 4-gray level image is shown at one pixel distance and at 0^0 orientation. CM can be defined as joint probability distribution of pixel values, when CM is normalized by the total number of possible pairs of pixels with fixed distance and orientation.

Table 2.2. Abbreviations for Haralick's features presented in Table 2.3.

N = Number of gray(wavelet) levels of the image $P(a,b)= CM_{\theta}^{d}(a,b)$

$$p_x(a)=\sum_{b=1}^{N} p(a,b), \qquad p_y(b)=\sum_{a=1}^{N} p(a,b),$$

$$\mu_x=\sum_{a=1}^{N} p_x(a), \qquad \mu_y=\sum_{a=1}^{N} p_y(a),$$

$$p_s(c)=\sum_{a=1}^{N}\sum_{b=1}^{N} p(a,b), \qquad c=a+b=2,3....2N$$

$$p_d(c)=\sum_{a=1}^{N}\sum_{b=1}^{N} p(a,b), \qquad c=a-b=0,1....N-1$$

$$\sigma_x=\sqrt{\sum_{a=1}^{N} p_x(a)(a-\mu_x)^2}, \qquad \sigma_y=\sqrt{\sum_{a=1}^{N} p_y(a)(a-\mu_y)^2}$$

$$HXY=-\sum_{a=1}^{N}\sum_{b=1}^{N} p(a,b)\log(p(a,b))$$

$$HXY1=-\sum_{a=1}^{N}\sum_{b=1}^{N} p(a,b)\log\left[p_X(a)p_y(b)\right]$$

$$HXY2=-\sum_{a=1}^{N}\sum_{b=1}^{N} p_X(a)p_y(b)\log\left[p_X(a)p_y(b)\right]$$

$$HX=-\sum_{a=1}^{N}\sum_{b=1}^{N} p_x(a)\log p_x(a)$$

$$HY=-\sum_{a=1}^{N}\sum_{b=1}^{N} p_y(a)\log p_y(a)$$

$$Q(a,b)=\sum_{c=1}^{N}\frac{p(a,c)p(b,c)}{p_x(a)p_y(b)} \qquad Q=\left[Q(a,b)\right]_{NXN}$$

Table 2.3. Haralick's Textural Features.

Feature	Definition
Angular second momentum	$ASM=\sum_{a=1}^{N}\sum_{b=1}^{N}(p(a,b))^2$
Contrast	$CON=\sum_{a=1}^{N}\sum_{b=1}^{N}(a-b)^2 p(a,b)$
Correlation	$COR=\frac{1}{\sigma_x \sigma_y}\sum_{a=1}^{N}\sum_{b=1}^{N}((ab)p(a,b)-\mu_x \mu_y)$
Sum of squares: Variance	$SSV=\sum_{a=1}^{N}\sum_{b=1}^{N}(a-\mu)^2 p(a,b)$
Inverse difference moment (Homogeneity)	$IDM=\sum_{a=1}^{N}\sum_{b=1}^{N}\frac{1}{1+(a-b)^2}p(a,b)$
Sum average	$SAv=\sum_{a=2}^{2N} a p_s(a)$
Sum variance	$SVa=\sum_{a=2}^{2N}(a-SAv)p_s(a)$
Sum entropy	$SEn=-\sum_{a=2}^{2N} p_s(a)\log(p_s(a))$
Entropy	$ENT=-\sum_{a=1}^{N}\sum_{b=1}^{N} p(a,b)\log(p(a,b))$
Difference variance	$DVA=\sum_{a=0}^{N-1} p_d(a)\left[a-\sum_{b=o}^{N-1} b(p_d(b))^2\right]$
Difference entropy	$DEn=-\sum_{a=0}^{N-1} p_d(a)\log(p_d(a))$
Information measure of correlation(1)	$IC1=\frac{HXY-HXY1}{\max\{HX,HY\}}$
Information measure of correlation(2)	$IC2=\left(1-\exp\left[-2.0\left\lvert HXY2-HXY\right\rvert\right]\right)^{1/2}$
Max. correlation coefficient	$MCC=(\text{second largest eigenvalue of } Q)^{1/2}$

Since CM is a square matrix whose dimension equals to the number of the gray levels, the number of the gray levels is proportional to the computational complexity.

Hence it is necessary to requantize the gray levels of the discrete wavelet images to decrease computational complexity. In this work, as the sea texture is more homogeneous than other natural textures, detail images are requantized to 16 gray levels (0-15) by a linear transformation. All CMs which are computed at 0^0, 45^0, 90^0, 135^0 directions are averaged so as to ensure the rotation invariance.

Haralick suggests 14 features describing textures using CM (Haralick et al.73). These features are shown in Table 2.3.

2.2.2. Feature Selection for Cooccurrence Features

Feature selection is a crucial task to reduce the dimensionality of feature vector because it is well known in pattern recognition that classification performance does not usually increases and the computational complexity of classification increases as the dimensionality of feature vector increases. If all appropriate features which are extracted for a classification problem and a finite number of training samples are given, there exists an optimal set of features that gives optimal classification performance and computational complexity for the task. Therefore, discrimination capacity of all features which are candidates for the optimal feature subset should be investigated. The problem in the feature selection algorithms is how to deal with large number of possible combinations of the feature subsets. Given a feature set $X_{k,}$ k=1,….,M , in order to find the best subset of m<M features that satisfies the selection criteria, the search is the combination problem that needs a selection from N possible solutions for a particular subset of m features. Instead of dealing with N combinations, a simpler approach based on determining an importance measure for each individual feature and ranking the features based on their importance is adopted here.

Fisher's linear discriminant analysis (FLDA) is a fundamental and widely used technique for determining discriminating power of the features. FLDA looks for directions in feature space that are efficient for discriminating samples in different classes. In FLDA, it is supposed that d-dimensional data is projected onto a line. Even if training samples are well-separated, projection onto an arbitrary line will generally produce poor clustering performance. However, by turning the line around, an orientation on which the samples are well-separated may be found. Two examples of

projection lines showing different orientations and different discrimination performances are seen in Figure 2.6. This is the main goal of linear discriminant analysis.

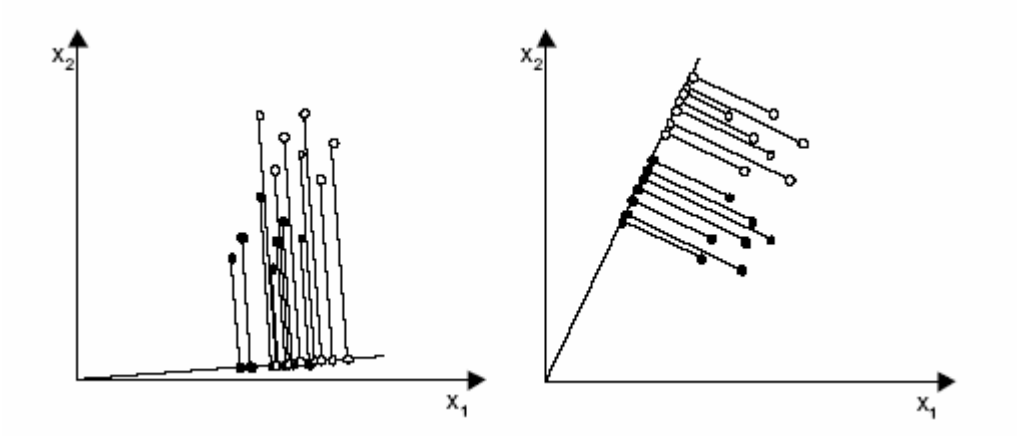


Figure 2.6. Projection of the same set of samples onto two lines in different directions. Figure on the right shows better separation of black and white points.

It is supposed that there is a set of n training samples, which have d-dimensional feature vectors, x_1,……, x_n, n_1 in the subset c1 and n_2 in the subset c_2.for the two-class case. This can be represented in the form of a linear combination of the components of $\underline{x}$ and by obtaining the scalar dot product

$$y = \underline{w}^t \underline{x}. \tag{2.18}$$

The corresponding set of n samples y_1,..., y_n are separated into the subsets Y_1 and Y_2. Each y_i is the projection of the corresponding x_i onto a line in the direction of $\underline{w}$.

In order to find the best projection vector $\underline{w}$, a measure of the separation between the projections could be the difference of sample means. The sample means and the means for the projected points are given by:

$$\underline{\mu}_i = \frac{1}{n_i} \sum_{\underline{x} \in D_i} \underline{x} \tag{2.19}$$

$$m_i = \frac{1}{n_i} \sum_{y \in Y_i} y = \frac{1}{n_i} \sum_{\underline{x} \in D_i} \underline{w}^t \underline{x} = \underline{w}^t \underline{\mu}_i \tag{2.20}$$

where m_i is the projection of μ_i and the distance between the projected means can be the criterion function:

$$J(\underline{w}) = |m_1 - m_2| = |\underline{w}^t (\underline{\mu}_1 - \underline{\mu}_2)|. \tag{2.21}$$

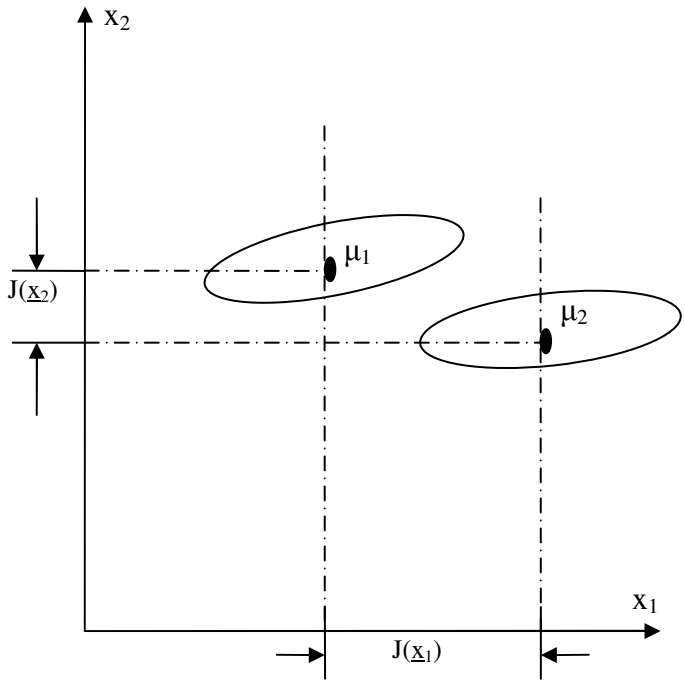


Figure 2.7. Axis x_2 yields better separation although axis x_1 has larger distance of projected means.

However, the distance between projected means is not a very good measure as shown in Figure 2.7 since it does not take into account the standard deviation within the classes. So Fisher's linear discriminant analysis is defined as the linear function $\underline{w}^t \underline{x}$

that maximizes the criterion function,

$$J(\underline{w}) := \frac{\left|m_1 - m_2\right|^2}{\tilde{s}_1^2 + \tilde{s}_2^2} \qquad \text{where } \tilde{s}_i^2 := \sum_{y \in Y_i} (y - m_i)^2 \tag{2.22}$$

$\tilde{s}_i^{\,2}$ is defined as within-class scatter and an estimate of the variance the pooled samples. FLDA looks for a projection where samples from the same class are projected very close to each other and, at the same time, the distance between the projected means are as large as possible as in Figure 2.8.

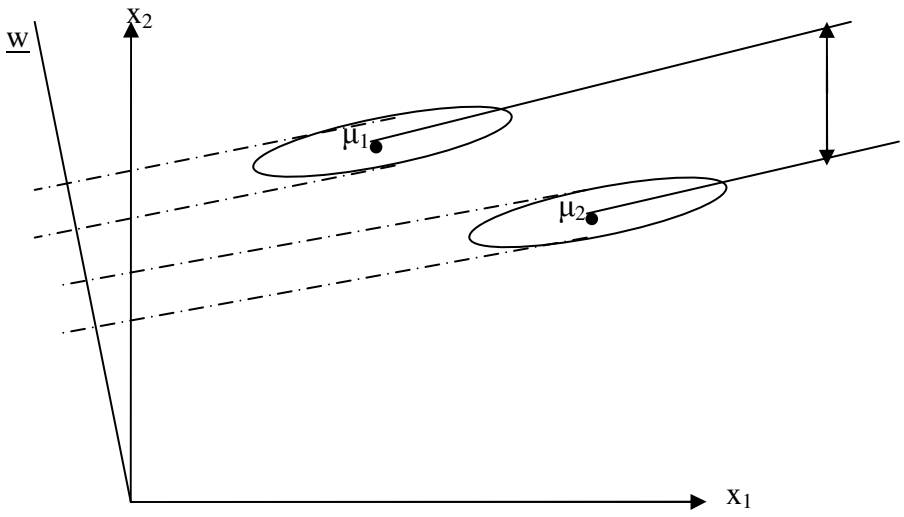


Figure 2.8. Vector w maximizing $J(\underline{w})$ obtains the best separation between the two sets of projected samples.

It is necessary that $J(\underline{w})$ is expressed as an explicit function of $\underline{w}$. For this reason the scatter matrix $\underline{S}_i$ is defined by,

$$\underline{S}_i := \sum_{x \in D_i}^{n} (\underline{x} - \underline{\mu}_i)(\underline{x} - \underline{\mu}_i)^t \tag{2.23}$$

and then scatter by:

$$\begin{aligned}\tilde{s}_i^{\,2} &:= \sum_{y\in Y}^{n}\left(y-m_i\right)^2 = \sum_{x\in D_i}^{n}\left(\underline{w}^t\underline{x}-\underline{w}^t\underline{\mu}_i\right)^2 \\ &= \sum_{x\in D_i}^{n}\underline{w}^t(\underline{x}-\underline{\mu}_i)(\underline{x}-\underline{\mu}_i)^t\underline{w} = \underline{w}^t\underline{S}_i\underline{w}.\end{aligned} \tag{2.24}$$

For the two-class case

$$\tilde{s}_1^{\,2}+\tilde{s}_2^{\,2} = \underline{w}^t\underline{S}_w\underline{w} \qquad \text{where} \quad \underline{S}_w = \underline{S}_1+\underline{S}_2. \tag{2.25}$$

$\underline{S}_w$ is called as *the within-class scatter matrix*. Similarly,

$$\begin{aligned}(m_1-m_2)^2 &= (\underline{w}^t\underline{\mu}_1-\underline{w}^t\underline{\mu}_2) = \underline{w}^t(\underline{\mu}_1-\underline{\mu}_2)(\underline{\mu}_1-\underline{\mu}_2)^t\underline{w} \\ &= \underline{w}^t\underline{S}_B\underline{w} \quad \text{where} \quad \underline{S}_B = (\underline{\mu}_1-\underline{\mu}_2)(\underline{\mu}_1-\underline{\mu}_2)^t.\end{aligned} \tag{2.26}$$

$\underline{S}_B$ is called as *the between-class scatter matrix* and the criterion function can be defined as

$$J(\underline{w}) := \frac{\underline{w}^t\underline{S}_B\underline{w}}{\underline{w}^t\underline{S}_w\underline{w}}. \tag{2.27}$$

In order to find vector $\underline{w}$ which maximize the criterion function, derivative of $J(\underline{w})$ must be equated to zero and one can derive following equation by taking derivation of $J(\underline{w})$:

$$\frac{d(J(\underline{w}))}{d\underline{w}} = \frac{\left(\frac{d}{d\underline{w}}\underline{w}^t\underline{S}_B\underline{w}\right)\underline{w}^t\underline{S}_W\underline{w} - \left(\frac{d}{d\underline{w}}\underline{w}^t\underline{S}_W\underline{w}\right)\underline{w}^t\underline{S}_B\underline{w}}{\left(\underline{w}^t\underline{S}_W\underline{w}\right)^2}$$

$$\frac{d(J(\underline{w}))}{d\underline{w}} = \frac{\left(2\underline{S}_B\underline{w}\right)\underline{w}^t\underline{S}_W\underline{w} - \left(2\underline{S}_W\underline{w}\right)\underline{w}^t\underline{S}_B\underline{w}}{\left(\underline{w}^t\underline{S}_W\underline{w}\right)^2} = 0$$

$$\left(\underline{S}_B\underline{w}\right)\underline{w}^t\underline{S}_W\underline{w} - \left(\underline{S}_W\underline{w}\right)\underline{w}^t\underline{S}_B\underline{w} = 0 \tag{2.28}$$

and by dividing (2.28) by $\underline{w}^t\underline{S}_w\underline{w}$,

$$\frac{\left(\underline{S}_B\underline{w}\right)\cancel{\left(\underline{w}^t\underline{S}_W\underline{w}\right)}}{\cancel{\left(\underline{w}^t\underline{S}_W\underline{w}\right)}} - \left(\underline{S}_W\underline{w}\right)\frac{\underline{w}^t\underline{S}_B\underline{w}}{\underline{w}^t\underline{S}_W\underline{w}} = 0$$

$$\left(\underline{S}_B\underline{w}\right) = \lambda\left(\underline{S}_W\underline{w}\right) \quad \text{where } \lambda = \frac{1}{J(\underline{w})} = \frac{\underline{w}^t\underline{S}_B\underline{w}}{\underline{w}^t\underline{S}_W\underline{w}}.$$

If inverse of $\underline{S}_w$ exists we can convert the problem to a standard eigenvalue problem by multiplying both sides by $\underline{S}_w^{-1}$,

$$\underline{S}_W^{-1}\underline{S}_B\underline{w} = \lambda\underline{w}. \tag{2.29}$$

But $\underline{S}_B\underline{x}$ for any vector $\underline{x}$ is in same direction as $\mu_1 - \mu_2$ such that:

$$\underline{S}_B\underline{x} = (\underline{\mu}_1 - \underline{\mu}_2)(\underline{\mu}_1 - \underline{\mu}_2)^t\underline{x} = \alpha(\underline{\mu}_1 - \underline{\mu}_2)$$
$$\text{where} \quad \alpha = (\underline{\mu}_1 - \underline{\mu}_2)^t\underline{x}. \tag{2.30}$$

Thus we can solve the eigenvalue problem immediately, vector $\underline{w}$ can be found by the equation,

$$\underline{w} = \arg\max_{\underline{w}} \left(J(w)\right) = \underline{S}_{\underline{w}}^{-1}(\underline{\mu}_1 - \underline{\mu}_2). \tag{2.31}$$

Coefficients of vector $\underline{w}$ correlate with the importance of features in discriminating samples into two classes. For this reason these coefficients can be used in feature selection. For the selection of the best feature subset from the whole feature set, importance of the features is calculated by:

$$I_k = \left| w_k \left(x_{k1} - x_{k2} \right) \right| \tag{2.32}$$

where I_k is the importance of the kth feature; w_k is the coefficient of kth feature and x_{km} is the mean of kth feature for the mth class (m=1,2). And then relative importance of the feature is calculated by normalizing the importance of the features,

$$R_k = \frac{I_k}{\sum_{k=1}^{n} I_k} \tag{2.33}$$

and they are ranked to discard the features whose relative importance is below a certain threshold. Thus selected features form a minimal feature set that provides best discrimination of the training samples for a particular classification task.

In this work, Haralick's textural features for the sea and land texture samples which are formed as 32X32, 16X16, 8X8, 4X4 and 2X2 square blocks have been computed. Then, relative importance of the each feature has been calculated from samples by using FLDA and ranked due to their values. Features with small relative importance value (RIV) (i.e. <0.1), have been discarded.

Table 2.4 Results for feature selection of the coocurrence features.

	32X32	16X16	8X8	4X4	2X2
Total number of sample blocks	4308	10110	14062	16302	16488
Difference variance(DVA)	0,88	0,87	0,82	0,72	0,82
Sum variance(SVa)	0,11	0,12	0,16	0,27	0,17
Angular second momentum(ASM)	~~0,0006~~	~~0,0005~~	~~0,001~~	~~0,00078~~	~~0,0015~~
Sum of squares: Variance(SSV)	~~0,0003~~	~~0,0008~~	~~0,003~~	~~0,0012~~	~~0,0037~~
Difference entropy(DEn)	~~0,0002~~	~~0,00022~~	~~0,001~~	~~0,00077~~	~~0,0012~~
Inverse difference moment : Homogeneity (IDM)	~~0,00016~~	~~0,00076~~	~~0,004~~	~~0,0036~~	~~0,0039~~
Sum entropy(SEn)	~~0,00004~~	~~0,00010~~	~~0,00019~~	~~0,0017~~	~~0,00019~~
Correlation(COR)	~~0,000035~~	~~0,000022~~	~~0,00004~~	~~0,00019~~	~~0,00004~~
Sum average(SAv)	~~0,000016~~	~~0,00003~~	~~0,00011~~	~~0,00023~~	~~0,00011~~
Entropy(ENT)	~~0,0000052~~	~~0,000004~~	~~0,000012~~	~~0,00001~~	~~0,00001~~
Contrast(CON)	~~0,0000013~~	~~0,000003~~	~~0,000007~~	~~0,00002~~	~~0,00001~~
Max. correlation coefficient(MCC)	~~0,0000009~~	~~0,0000002~~	~~0,000001~~	~~0,0000013~~	~~0,000001~~
Information measure of correlation(2)(IC2)	~~0,0000008~~	~~0,0000004~~	~~0,000005~~	~~0,0000081~~	~~0,000005~~
Information measure of correlation(1(IC1)	~~0,0000007~~	~~0,000001~~	~~0,000003~~	~~0,0000085~~	~~0,000003~~

The results of the feature selection of the coocurrence features are given in Table 2.4 . Here difference variance and sum variance have been selected as cooccurrence features.

Several coastline images were segmented by means of selected features by dividing them to 32X32, 16X16, 8X8, 4X4 and 2X2 square blocks. Although selected features show good segmentation performance for 32X32 and 16X16 blocks, they give poorer performance for smaller blocks. An example of coastline image segmentation with selected CM features is shown in Figure 2.9.

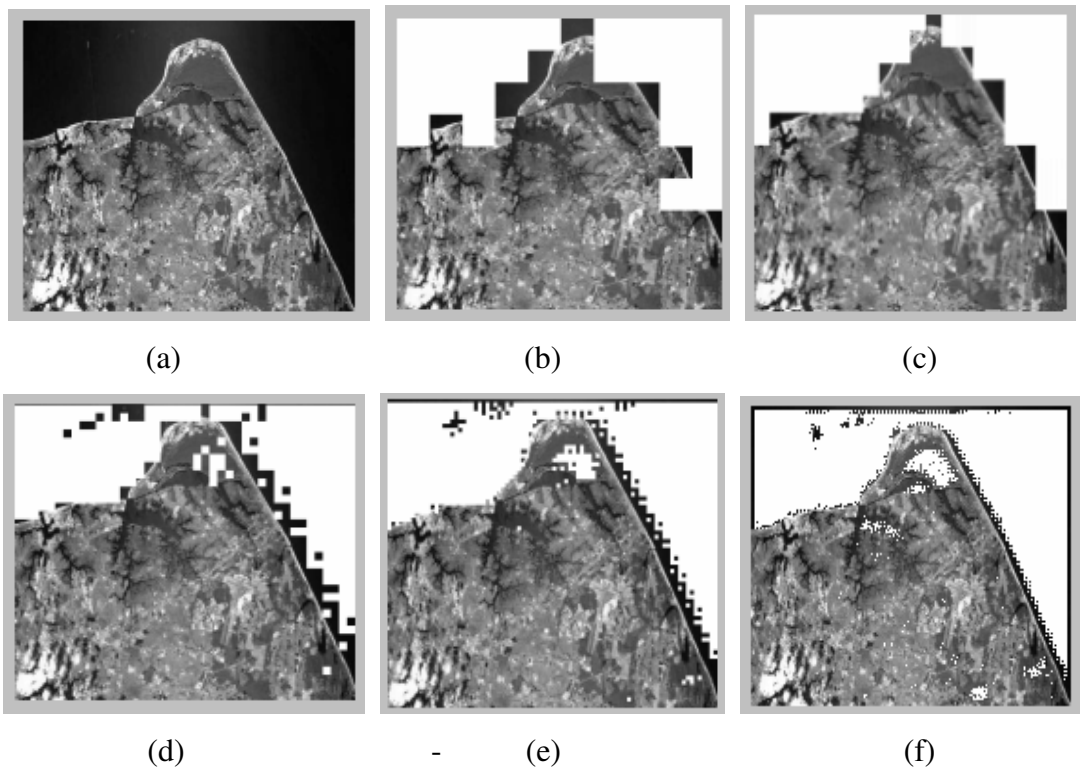


Figure 2.9. Segmentation by CM features. (a) Coastline image (b) 32X32 (c) 16X16 (d) 8X8 (e) 4x4 (f) 2X2 block segmentations.

2.2.3. Histogram Features

By means of CM features, segmentation of a coastline image which is divided into large blocks give good results because they describe the periodicity of the textures. However, accuracy of the segmentation decreases and computational cost increases since the periodicity of the texture vanishes as the block size get smaller since second order statistics cannot capture periodicity. Another problem is the increased computational complexity as the number of blocks is increased. For segmentation of a coastline image by smaller blocks, it is convenient to employ histogram features since the histogram of texture describes the frequency of gray values rather than spatial dependence of the gray values.

Using image histogram directly for feature extraction is important since it yields translation invariant features. However in order to get more stable texture characterization higher order moments of image histograms should be computed. Instead Mallat (Mallat 1989) found experimentally that detail histograms of natural

textured wavelet images can be modeled with the family of exponentials:

$$h(u) := Ke^{-(|u|/\alpha)^{\beta}}. \tag{2.34}$$

The parameter β modifies the decreasing rate of the peak and α models the variance. The constant K is selected such that $\int h(u)du = 1$. The model parameters α, β and K are calculated by

$$m_1 = \int |u| h(u)du \qquad m_2 = \int |u|^2 h(u)du. \tag{2.35}$$

Substituting (2.34) and changing the variables in two integrals (2.35) parameters are obtained by

$$K = \frac{\beta}{2\alpha\Gamma(1/\beta)} \quad \text{where} \quad \Gamma(x) = \int_0^{\infty} e^{-u}u^{x-1}du \tag{2.36}$$

$$\alpha = m_1 \frac{\Gamma(1/\beta)}{\Gamma(2/\beta)} \tag{2.37}$$

$$\beta = F^{-1}\left(\frac{m_1^{\,2}}{m_2}\right) \quad \text{where } F(x) = \frac{\Gamma^2(2/x)}{\Gamma(1/x)\Gamma(3/x)}\ . \tag{2.38}$$

Since model parameter K is a constant and α, β have independent characteristics of detail image, α and β can be selected as the histogram features of detail image. the function $F^{-1}(x)$ is given in Figure 2.10.

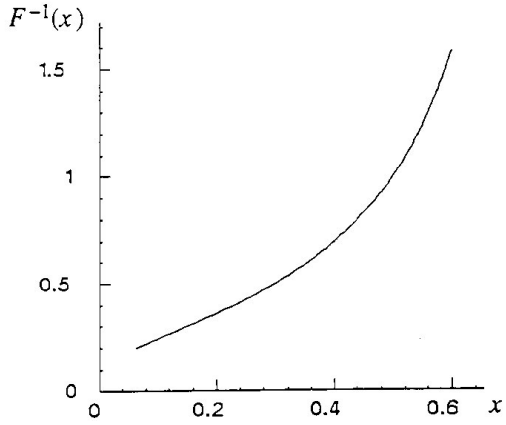
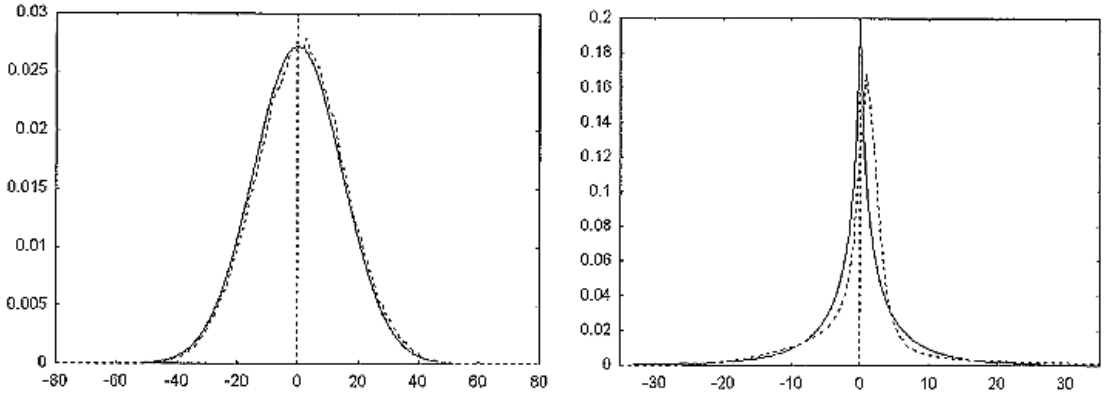


Figure 2.10. Graph of the function $F^{-1}(x)$ characterized by (2.38). (Source: (Mallat 1989))

The typical examples of detail image histograms obtained from the wavelet representation of the real images and the graphs of the model obtained from (2.34) are shown in Figure 2.11.

In this work, several coastline images were segmented by means of selected features (α and β) by dividing them to 32X32, 16X16, 8X8, 4X4 and 2X2 square blocks. Selected features show good segmentation performance for 8X8, 4X4 and 2X2 blocks.



(a) (b)

Figure 2.11.Examples of the model. The dotted line is the observed detail histogram; the solid line is the fitted model. (a) α=20.8 and β=2.0. (b) α=1.27 and β=0.566. (Source: (Mallat 1989))

An example of coastline image segmentation with histogram features is shown in Figure 2.12.

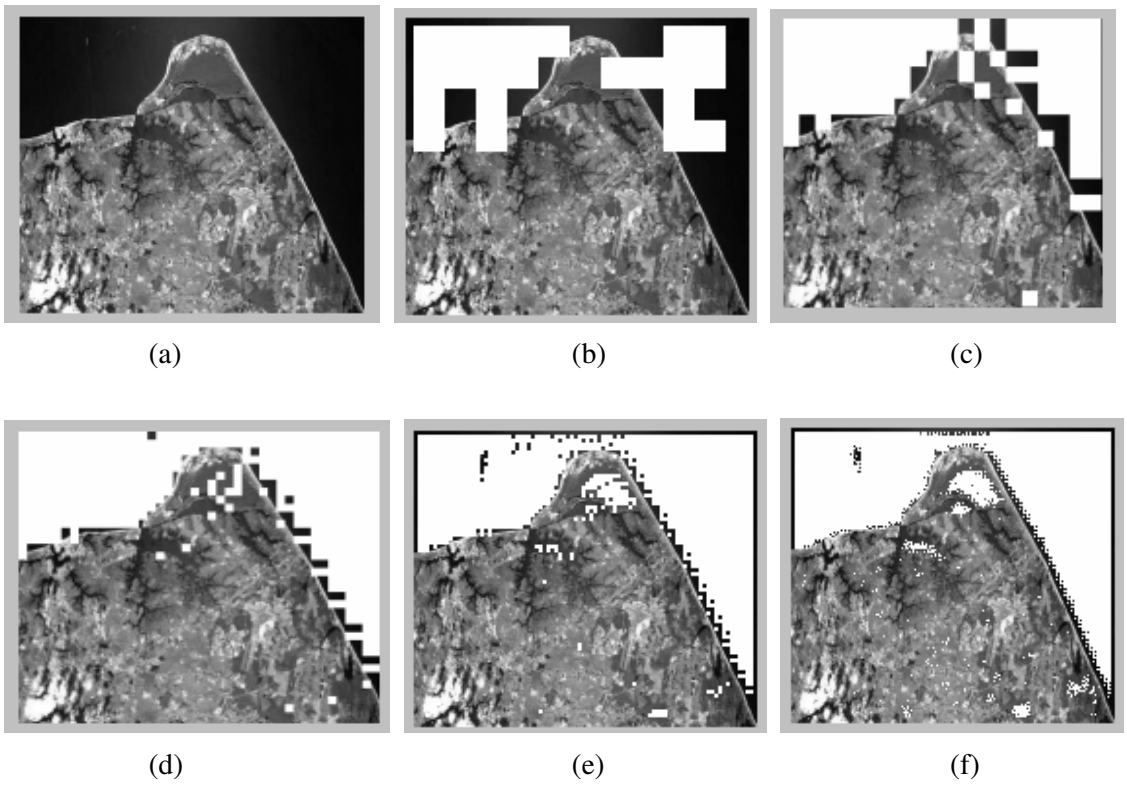


Figure 2.12.Segmentation by histogram features (a) Coastline image (b) 32X32 (c) 16X16 (d) 8X8 (e) 4X4 (f) 2X2 block segmentations

2.3. Maximum Likelihood Classifier

The maximum likelihood (ML) classifier is one of the most widely used classifiers in pattern recognition. ML classifier has several advantages over the other supervised classifiers. ML parameters converges as the number of training samples increases and it is often simpler and more accurate than other classifiers. It is conjectured that the features which have been extracted for describing natural textures extracted from aerial images are jointly Gaussian distributed (Thyagarajan et al. 1994). For those reasons, a maximum likelihood classifier can be employed for classification of sea texture. As it is supposed that each class contains N samples $D=\{x_1,...,x_N\}$ and samples from different classes (sea or land) are statistically independent, the joint pdf can be written as:

$$p(D|\underline{x}_k) = \prod_{k=1}^{N} p(\underline{x}_k|D). \tag{2.39}$$

The maximum likelihood estimate of $\underline{\theta}$ is, by definition, the value that maximizes $p(D|\underline{\theta})$. For the simplicity it is convenient to use the loglikelihood:

$$l(\theta) := \log[p(D|\underline{x}_k)] \tag{2.40}$$

$$l(\theta) := \sum_{k=1}^{N} \log[p(\underline{x}_k|D)]. \tag{2.41}$$

since $\underline{\theta}$ maximizes the loglikelihood, by taking derivative of loglikelihood function with respect to $\underline{\theta}$,

$$\nabla_\theta l = \sum_{k=1}^{N} \nabla_\theta \log[p(\underline{x}_k|\theta)] \tag{2.42}$$

and by setting $\nabla_\theta l=0$, ML estimate of $\underline{\theta}$ can be found. In this particular case where the feature vectors are jointly Gaussian distributed, the parameter vectors are the mean $\underline{\mu}$ and covariance $\underline{\Sigma}$ of the pdf of the feature vectors and these parameters are estimated from samples:

$$\underline{\mu} = \frac{1}{N}\sum_{k=1}^{N} \underline{x}_k \tag{2.43}$$

$$\underline{\Sigma} = \frac{1}{N}\sum_{n=1}^{N} (\underline{x}_n - \underline{\mu})(\underline{x}_n - \underline{\mu})^t. \tag{2.44}$$

After the parameters are estimated, the loglikelihood function for each class of textures

can be given by (Duda et al. 2001):

$$\begin{aligned}\Lambda_k &= \log\{p(\underline{x}\,|\,\underline{\mu}_k, \underline{\Sigma}_k)\} \\ &= -\frac{1}{2}\log\{(2\pi)^d |\underline{\Sigma}_k|\} - \frac{1}{2}(\underline{x}-\underline{\mu}_k)^t \underline{\Sigma}_k^{-1}(\underline{x}-\underline{\mu}_k)\end{aligned} \tag{2.45}$$

where $\underline{x}$ is the feature vector and d is the feature vector dimension.

If loglikelihood of sea texture is greater than that of land texture, then unknown texture is classified as sea otherwise it is classified as land.

2.4. Multiscale Segmentation

In texture segmentation, the size of classification window is very important. In general, employing a large window increases the accuracy of the classification since it has rich statistical information but at the same time, it may contain pixels from different classes. A large window increases the accuracy of the segmentation in large and homogeneous regions such as sea region but it results in bad segmentations near the boundaries between the different textures. A small window contains fewer pixels from different classes but it decreases the accuracy of the classification because of less statistical information. So it is reasonable to employ large windows at the beginning for acquiring the coarse segmentation and then to employ smaller windows for refining the large blocks near the boundary between different classes.

In this thesis, a quad-tree approach has been employed for implementing the refinement process by using different window sizes for segmentation. In quad-tree approach, the image is considered as an initial 2^J X 2^J square image, the windows are obtained by dividing the image into four square images which are same size as in Figure 2.13.

In this work, the image is initially divided into nonoverlapping regions of 32X32 size in order to start to classification with 32X32 windows. Firstly, the image is classified by using CM features as sea and land regions and then sea regions

neighboring land regions or land regions neighboring sea regions are marked as boundary regions between sea and land regions as shown Figure 2.14. For the next step, all regions are divided into four “child“ 16X16 subregions and they are classified due to the class of their parent regions. If the class of the parent region is sea or land, the class of the subregion is initially same as the class of the parent region. If the parent region is boundary region, the subregions are classified by using CM features.

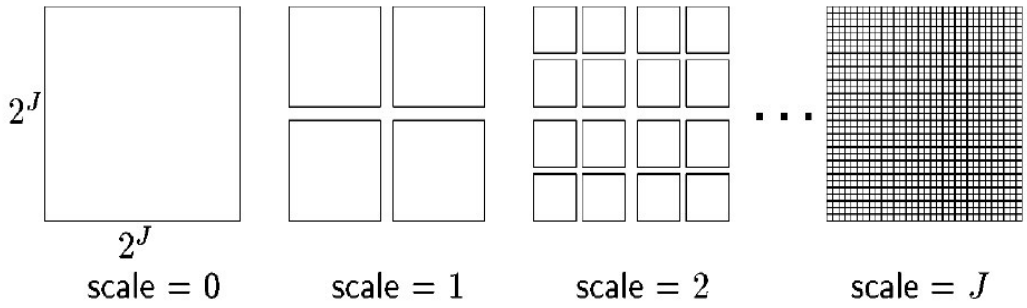


Figure 2.13. Image divided into windows at different scales.

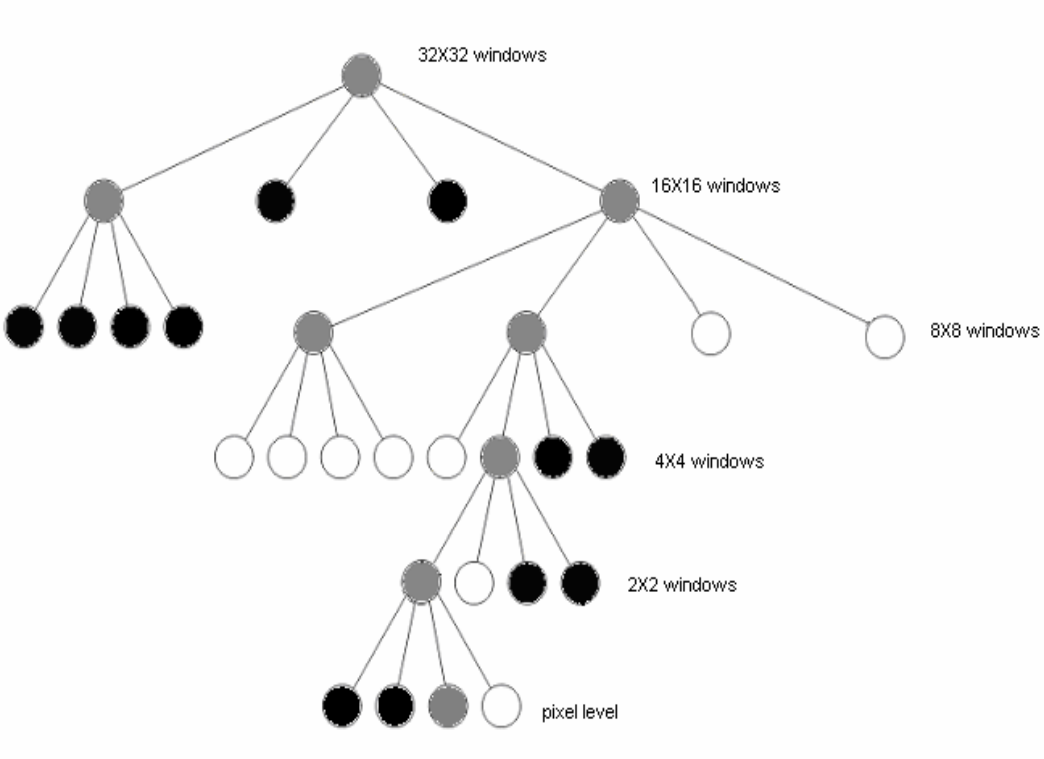


Figure 2.14. Quad-tree structure of the windows. White, black and gray circles represent sea, land and boundary regions respectively.

After classification of the all subregions, initial boundary subregions are selected. Some of the boundary subregions may contain a parent region which is not a boundary region. In other words these subregions may be incorrectly classified by using CM features and they should be iteratively reclassified by using CM features. New boundary subregions should be selected until all boundary subregions are classified by using CM features.

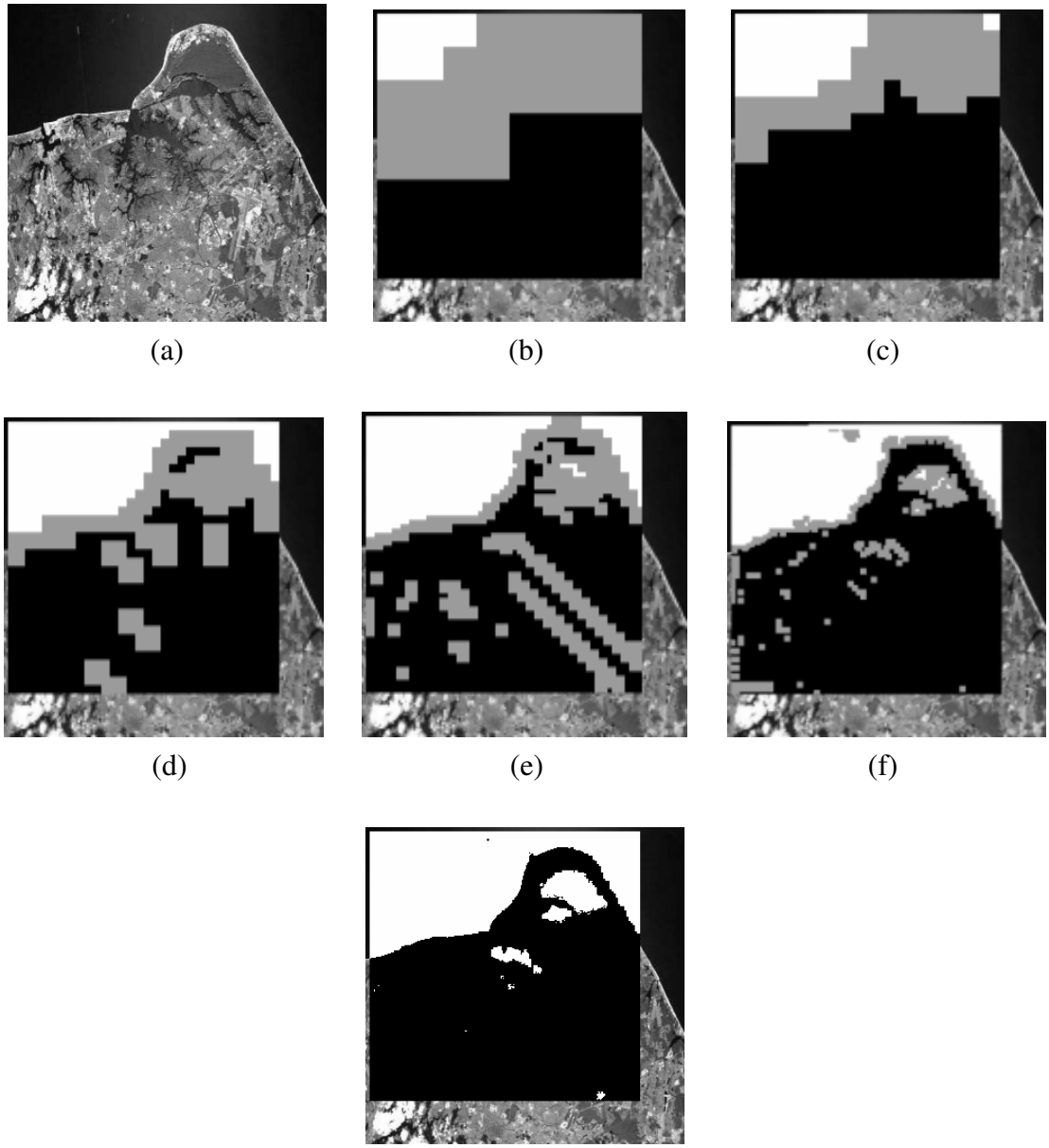


Figure 2.15.Multiscale Segmentation (a) Coastline image (b) 32X32 (c) 16X16 (d) 8X8 (e) 4x4 (f) 2X2 (g) Pixel Level segmentation.

For the 8X8, 4X4 and 2X2 window levels, same procedure is recursively applied but instead of CM features; histogram features are used and for the pixel level segmentation, energy and mean deviation features are used. An example of proposed multiscale coastline image segmentation is shown in Figure 2.15.

2.5. Postprocessing of Segmented Image

After the segmentation of the coastline image into sea and land textures, it is necessary to apply postprocessing operations in order to remove very small islands, lakes and noisy pixels from the image, to detect sea boundary pixels and to thin the boundary for getting one-pixel wide coastline and finally to store coordinates of coastline points.

In this thesis, median filter and morphological operators (dilation and erosion) are used to remove very small islands, lakes and noisy pixels from the image.

2.5.1. Median Filter

Median filtering is a non-linear image enhancement method for smoothing of signals by suppression of impulsive noise while preserving of edges. It replaces each pixel value with the median of the gray values in the local neighborhood.

Median filters are very robust in removing salt and pepper type of noise since they completely discard the values which are extremely different from the values in the neighborhood. Firstly, pixel values in NxN neighborhood are sorted into ascending order by their gray values. Then, the value of the middle pixel is selected to replace with the value of the pixel under consideration. An odd-size neighborhood is used for computing the median. However, if the number of pixels is even, the median is taken as the average of the middle two pixels after sorting.

2.5.2. Dilation and Erosion

Mathematical morphology contributes a wide range of operators to image processing, all based on a fundamental concepts from set theory. Translation of a binary image A by a pixel p shifts the origin of A to p. The translation of A by pixel p is an image defined by,

$$A_p := \{a + p \,|\, a \in A\}. \tag{2.46}$$

If A_{b1}, A_{b2}… A_{bn} are translations of the binary image A by the 1 pixels of the binary image B= {b_1, b_2… b_n}, then the union of the translations of A by the 1 pixels of B is called the **dilation** of A by B and is defined by

$$A \oplus B := \bigcup_{b_i \in B} A_{b_i}. \tag{2.47}$$

B is called structuring element. The size and shape of the structuring element determines the effect of the dilation on the input image A.

The dual of dilation is **erosion.** The erosion of a binary image A by a binary image B is at a pixel p if every 1 pixel in the translation of B to p is also 1 in A. Erosion is defined by

$$A \Theta B := \{p \,|\, B_p \subseteq A\}. \tag{2.48}$$

Erosion and dilation are usually used in order to filter images. If noise type is known, then suitable structuring element can be used and a sequence of erosion and dilation operations can be employed for removing the noise.

2.5.2.1. Thinning Algorithm

After the boundaries are detected, these boundaries are needed to be thinned in order to extract the shape information. The algorithm used here for thinning binary images is due to (Zhang and Suen 1984). Object points are assumed to have value 1 and background points to have value 0. This method involves successive passes of two basic steps applied to the contour points of the given region, where a contour point is any pixel with value 1 and having at least one 8-neighbor valued 0. With reference to the 8-neighborhood definition shown in Figure 2.16, step 1 flags a contour point p for deletion if the following conditions are satisfied:

$$\begin{aligned} &(a)\ 2 \le N(p_1) \le 6; \\ &(b)\ S(p_1) = 1; \\ &(c)\ p_2.p_4.p_6 = 0; \\ &(d)\ p_4.p_6.p_8 = 0; \end{aligned} \tag{2.49}$$

where $N(p_1)$ is the number of nonzero neighbors of p_1;

$$N(p_1) = p_2 + p_3 + \ldots\ldots\ldots\ldots + p_8 + p_9 \tag{2.50}$$

and $S(p_1)$ is the number of 0-1 transitions in the ordered sequence of $p_2, p_3, \ldots, p_8, p_9, p_2$.

p_9	p_2	p_3
p_8	p_1	p_4
p_7	p_6	p_5

Figure 2.16.Neighborhood representation used by the thinning algorithm. $p_2, p_3, \ldots, p_9$ are the 8-neighbors of p_1.

In step 2, conditions (a) and (b) remain same, conditions (c) and (d) are changed to

$$\begin{aligned}(c)\ p_2.p_4.p_8 &= 0;\\ (d)\ p_2.p_6.p_8 &= 0;\end{aligned} \tag{2.51}$$

Step1 is applied to every border pixel in the binary object under consideration. If one or more conditions are violated, the value of the point in question is not changed. If all conditions are satisfied the point is flagged for deletion. But, the point is not deleted until all border points have been inspected. After step 1 has been applied to all border points, those that were flagged are deleted (changed to 0). Then, step 2 is applied to the resulting image in exactly the same manner as step 1.

One iteration of the thinning algorithm consists of (1) applying step 1 to flag border points for deletion; (2) deleting the flagged points (3) applying step 2 to flag the remaining border points for deletion; (4) deleting the flagged points. This procedure is applied iteratively until no further points are deleted. At that time the algorithm terminates, resulting in the skeleton of the boundary.

2.5.3. Extraction of Coastlines

In the implementation of proposed automatic coastline extraction method, following steps are applied:

1. Aerial coastline image is transformed into wavelet image by using four-tap Daubechies FIR filters.
2. Wavelet image is first divided into 32X32 nonoverlapping bins. For each bin, the selected cooccurrence features: sum variance and difference variance, are computed. All bins are classified as sea bins or land bins by ML classifier. ML classifier is trained by employing the training samples of sea texture and land texture which are captured from 15 different coastline images.

3. Sea bins neighboring land bins and land bins neighboring the sea bins are marked for reclassification.
4. Wavelet image is divided into 16X16 nonoverlapping bins by dividing 32x32 bins into four 16X16 bins. If parent bin of 16X16 bin are the selected bin for reclassification, then cooccurrence features of the bin are computed and it is reclassified by ML classifier, else class of 16X16 bin remain same as that of the parent bin .
5. Sea bins neighboring land bins and land bins neighboring the sea bins are marked for reclassification.
6. Wavelet image is divided into 8X8 nonoverlapping bins by dividing 16x16 bins into four 8X8 bins. If parent bin of an 8X8 bin are the selected bin for reclassification, then histogram features, α and β, of the bin are computed and it is reclassified by ML classifier, else class of 8X8 bin remain same as that of the parent bin.
7. Sea bins neighboring land bins and land bins neighboring the sea bins are marked for reclassification.
8. Step 6 and 7 are repeated for 4X4 and 2X2 bins.
9. Wavelet image is divided into pixels by dividing 2x2 bins into four pixels. If parent bin of the pixel are the selected bin for reclassification, then energy and mean deviation features of the bin are computed in 3X3 pixel neighborhood and it is reclassified by ML classifier, else class of the pixel remain same as that of the parent bin. Thus, coastline image is segmented into sea and land regions.
10. A binary image is obtained by replacing values of sea pixels with 255 and values of land pixels with 0.
11. In order to remove very small islands, lakes and noisy pixels from the image, 5X5 median filter is first applied to the binary image two times. Then, 3X3 opening (erosion followed by dilation) operators are applied, sequentially.
12. Boundary of sea region is simply detected as coastline by selecting land pixels which are neighbor to sea pixels or by selecting sea pixels neighboring land pixels.

13. Detected boundaries are thinned in order to get one-pixel-wide coastline by means of the thinning algorithm which is described in 2.5.2.1.

14. Coordinates of the longest coastline in the image is stored.

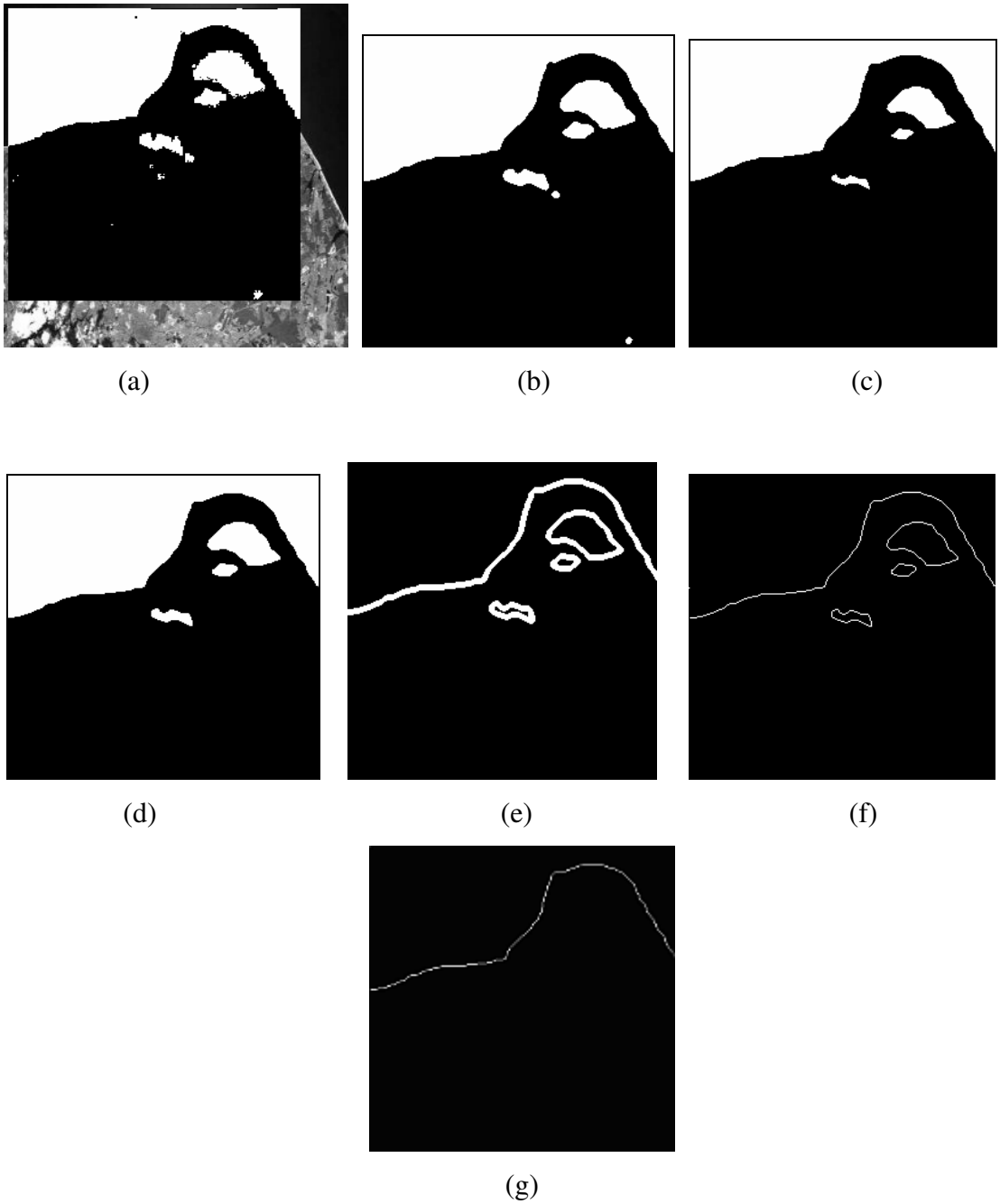


Figure 2.17. An example of results of postprocessing of segmented coastline image. (a)Segmented image (b) 5X5 Median filter (c) Erosion (d) Dilation (e) Boundary detection (f) Thinning (g) the longest coastline in the image

An example of results of postprocessing of segmented coastline image is shown in Figure 2.17.

Proposed coastline extraction method has been tested with more than thirty aerial coastline images. It extracted coastlines with a maximum four-pixel error except for the images that contains very noisy sea regions and very smooth land regions.

CHAPTER 3

SHAPE DESCRIPTORS

In the previous chapter, coastline extraction problem has been considered the same way as an object boundary detection problem. Similarly, we can consider coastline matching problem as an object boundary (shape) matching problem. For this reason, in this thesis research, we have studied various shape matching algorithms to develop a coastline matching algorithm. We have especially focused on the matching algorithms which are employed in content-based image retrieval (CBIR) applications, since CBIR is similar to the automatic coastline matching with map data in that CBIR looks for the query object in image database where the automatic coastline matching looks for query coastline in map database.

In shape matching, a key issue is to extract the effective and perceptually important shape features depending on object boundary information. These features are derived from a choice of a shape representation (shape descriptors) and they are used for computing similarities between the objects. For good retrieval accuracy, an algorithm based on a shape descriptor must be able to find perceptually similar shapes from databases, even if they undergo geometric transformations, this means that the shape descriptor must be invariant for translated, rotated, and scaled shapes. The descriptors must be robust to noise, distortion and deformations which are tolerated by human beings. It must be compact for indexing the database and computationally efficient for real-time applications.

Shape representation and matching techniques (Zhang and Lu 2004) in the literature can be generally classified into two class which are *contour-based* and *region-based* methods. Contour-based techniques only use object boundary information whereas region based techniques use all pixels within the object to obtain the shape descriptors. In coastline matching, extracted coastlines from aerial images are mostly open-curved and whole object information about sea and land regions cannot be captured. For these reasons, region-based techniques which need whole information about the object are not suitable for coastline matching.

Among contour-based methods, simple global descriptors such as *area,*

circularity, eccentricity, and *major axis orientation* usually can only discriminate shapes with the large differences and ignore local deviations of the boundary which are important in coastline matching. Hausdorff distance is a correspondence-based shape descriptor using point-to-point matching. Hausdorff distance is not rotation and scale invariant and it is sensitive to noise. Chain code representations, Fourier descriptors, and curvature scale space descriptors are very robust shape matching methods in contour-based techniques. In this chapter, these methods will be explained and advantages, disadvantages for coastline matching will be discussed.

3.1. Chain Code Representations

Chain codes are used to represent an object boundary by a connected sequence of straight-line segments with specified length and orientation. Chain code representations are mostly used in handwritten character recognition applications (Gyeonghwan and Govindaraju 1997). The method was introduced by Freeman (Freeman 1961) who described a method permitting the encoding of arbitrary geometric configurations. Chain code representation is based on the 4-directional and 8-directional segments as shown in Figure 3.1.

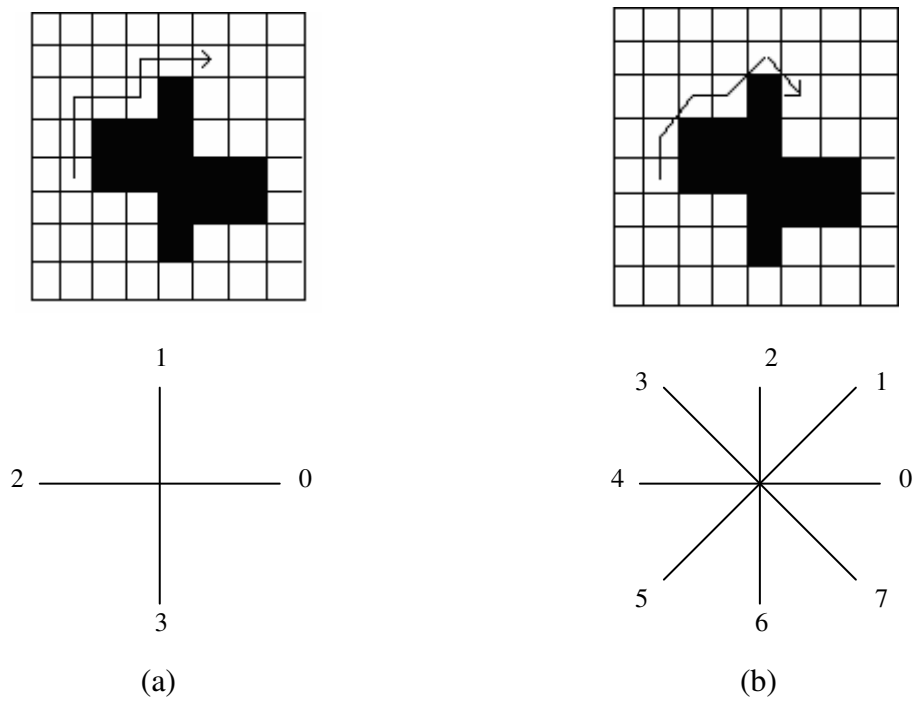


Figure 3.1. Directions for (a) 4-directional chain code and (b) 8-directional chain code

If the chain code is used for matching, it must be invariant to transformations such as translation, rotation and scaling. Invariance property can be achieved by representing the differences in chain code in the successive directions. This can be implemented by subtracting each element of the chain code from the previous one and taking the result modulo N, where N is the connectivity (i.e. 8 or 4). Resulting representation is called *differential chain code representation* which is translation and rotation invariant. However, this modified chain code representation is not scale invariant.

Iivarinen and Visa (Iivarinen and Visa 1996) introduced *a chain code histogram* (CCH) for object recognition. CCH is computed as $p(k)=n_k/n$, where n_k is the number of chain code having k value in a chain code and n is the total number of chain code elements. The CCH reflects the probabilities of different directions in a boundary. The CCH is translation and scale invariant, but it is not rotation invariant. To achieve rotation invariance, the normalized CCH is proposed. It is defined as $p(k)=l_k n_k/l$, where n_k is the same as in CCH, l_k is the length of the direction k and l is the length of the contour. Thus, the normalized CCH is translation, rotation and scale invariant.

Chain code representation is sensitive to noise and variations on the object boundary. Matching of deformed object boundaries is almost impossible. CCH is invariant to transformations but it loses spatial locations of the object boundary which is important for open curve matching. For these reasons, chain code representation is not suitable for coastline matching.

3.2. Curvature Scale Space (CSS) Descriptors

CSS descriptors are widely used in shape representation. CSSD describes main local shape features. By representing shape boundary in scale space, not only the locations of convex (or concave) segments, but also the degree of convexity (or concavity) of the segments on the shape boundary are detected (Mokhtarian et al. 1996). In the development of MPEG-7 standardization (Jeannin 2000), CSS descriptor (CSSD) has been selected as the standard contour shape descriptor. It is used in various applications such as leaf classification (Mokhtarian and Abbasi 2004), multiview 3-D object recognition (Mokhtarian and Abbasi 2001), sketch-based image retrieval

(Matusiak et al. 1998), corner tracking (Mohanna and Mokhtarian 2002), hand pose recognition (Chin-Chen et al 2002), terrain-aided navigation (Madhavan et al 2002), and analysis of ECG waveform (Jager et al. 1990).

In this section, construction of the CSSD, matching algorithm and invariance to transformations of CSSD will be explained.

3.2.1. Construction of the CSSD

The curvature κ along a curve is defined by (Mokhtarian and Mackhworth 1992):

$$\kappa(s) := \lim_{\Delta s \to 0} \frac{\Delta\theta}{\Delta s} \tag{2.52}$$

where Δθ is the change in tangent angle produced by a change Δs of the arclength. From this definition it follows that the curvature κ at [x(u),y(u)] is computed by:

$$\kappa(u) = \frac{\dot{x}(u)\ddot{y}(u) - \ddot{x}(u)\dot{y}(u)}{(\dot{x}^2(u) + \dot{y}^2(u))^{3/2}} \tag{3.1}$$

where $\dot{x}(u)$, $\dot{y}(u)$ are first derivatives of x(u),y(u) and $\ddot{x}(u)$, $\ddot{y}(u)$ are second derivatives of x(u),y(u). Let g(u,σ), a 1-D Gaussian kernel of width σ, be convolved with each component of the curve, then let X(u,σ) and Y(u,σ) represent the components of the resulting curve, C_σ:

$$\begin{aligned} X(u,\sigma) &= x(u) * g(u,\sigma) \\ Y(u,\sigma) &= y(u) * g(u,\sigma) \end{aligned} \tag{3.2}$$

where " * " denotes convolution operator. Due to properties of the convolution, the derivatives of each component can be computed easily by:

$$\dot{X}(u,\sigma) = x(u) * \dot{g}(u,\sigma)$$
$$\dot{Y}(u,\sigma) = y(u) * \dot{g}(u,\sigma) \tag{3.3}$$

and,

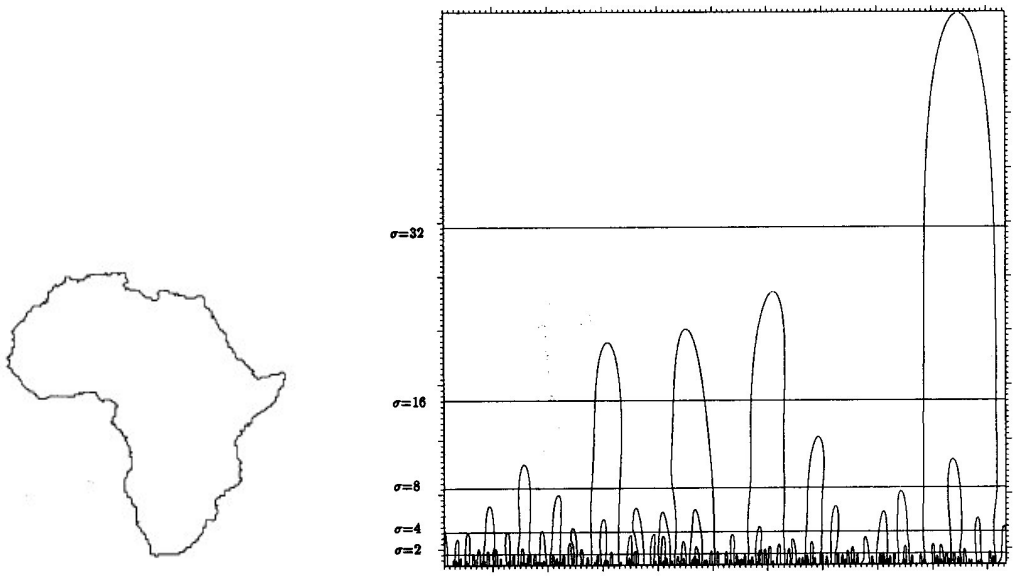
$$\ddot{X}(u,\sigma) = x(u) * \ddot{g}(u,\sigma)$$
$$\ddot{Y}(u,\sigma) = x(u) * \ddot{g}(u,\sigma) \tag{3.4}$$

where $\dot{g}(u,\sigma)$ and $\ddot{g}(u,\sigma)$ are the first and second derivatives of the gaussian kernel, $g(u,\sigma)$ with respect to u, respectively. Then, curvature of the curve can be computed as:

$$\kappa(u,\sigma) = \frac{\dot{X}(u,\sigma)\ddot{Y}(u,\sigma) - \ddot{X}(u,\sigma)\dot{Y}(u,\sigma)}{\left(\dot{X}^2(u,\sigma) + \dot{Y}^2(u,\sigma)\right)^{3/2}}. \tag{3.5}$$

The implicit function defined by $\kappa(u, \sigma) = 0$ is the CSS image of the curve, C(u). If the curvature zero crossings of evolved curve $C_\sigma(u)$ are computed during evolution, the resulting points can be displayed in the *(u, σ)* plane. CSS image of coastline of Africa is shown in Figure 3.2 as an example of CSS image. Here *u* is an approximation of normalized arclength and *σ* is the width of the Gaussian kernel. As *σ* increases, $C_\sigma(u)$ becomes smoother and number of zero crossings decreases. When σ becomes sufficiently large, $C_\sigma(u)$ will be a convex curve with no zero crossing and the process of evolution is stopped. The result of this process is represented by the binary CSS image of the curve. The small contours of CSS image represent the minor deviations on the boundary which may be noise and can be ignored. Each boundary can be effectively represented by the locations of the maxima of its CSS image contours. After extracting

the maxima of CSS image contours, they are normalized so that horizontal coordinate u varies in the range [0,1]. The normalized CSS maxima values are used as the feature vector of the CSS contour in the image.



(a) (b)

Figure 3.2. (a) Coastline of Africa, (b) CSS image of coastline of Africa. (Source: (Mokhtarian and Mackhworth 1992))

3.2.2. CSS Matching Algorithm

The complete CSS matching algorithm given in (Mokhtarian et al 2000) comparing the two sets of CSSDs, one from the image and the other from the model image in the image database is as following:

1. Create a node consisting of the largest scale CSSD of the image and the largest scale CSSD of the model. Initialize the cost of this node to absolute difference of σ-coordinates of the image and the model.
   Compute a CSS shift parameter $\alpha = U_m - U_i$ for a each node where U is the horizontal coordinate of a maximum of CSSDs, i and m refers to image and model, respectively. This parameter is used to compensate the effect of the different starting points and change in orientation.
2. If there are more than one maximum in the model having a σ-coordinate

close (within 80%) to the largest scale CSSD of the image, create extra nodes consisting of the largest scale CSSD of the image and respective additional CSSD of the model. Also create the same nodes for the second largest scale CSSD of the image and the respective CSSD of the model. Initialize the cost and compute the CSS shift parameter for each node accordingly.

3. Create two lists for each node obtained in step 1 and 2. The first list will contain the image curve CSSDs, and the second list will contain the model curve CSSDs matched in that node at any point of the matching procedure. Initialize the first and second lists of each node by the CSSDs determined in the first two steps.
4. Expand each node created in step 1 and 2 using the procedure described in step5.
5. To expand a node, select the largest scale CSSD of the image which is not in the first list and apply that node's shift parameter α to map that CSSD to the model. Locate the nearest CSSD of the model which is not in the second list. If the two CSSDs are separated by a reasonable horizontal distance (0.2 of the maximum possible distance) define the cost of the match as the straight line distance between the two CSSDs. Otherwise, define the height of the CSSD as the cost of the match. If there are no more remaining CSSDs of the image, define the cost of the match as the height of the highest CSSD of the model not in the node's second list. Likewise, if there are no more remaining CSSDs of the model, define the cost of the match as the height of the selected CSSD of the image. Add the match cost to the node cost. Update the two lists with the node.
6. Select the lowest cost node. If there are no more CSSD of the model or the image that remain unmatched within that node, then return that node as the lowest cost node. Otherwise, go to step 5 and expand the lowest cost node.
7. Reverse the place of the image and the model and the repeat the steps 1-6 to find the lowest cost in this case.
8. Consider the lowest node as final matching cost between the image and the model.

The algorithm should be repeated once more for the mirror image of the image. The final matching cost is selected as the lower value of the matching costs of the image and the mirror image.

3.2.3. Invariance Properties of CSSDs with respect to Transformations

The core of the CSSDs is the curvature zero crossing points. These points are translation invariant. Scale invariance is achieved by normalizing CSSDs so that horizontal coordinate u varies in the range [0, 1]. Rotation invariance is achieved by mapping the CSSDs of the image to those of the model using α shift parameter described in 3.2.2.

For closed boundaries, CSSD has several attractive properties. It is perceptually meaningful and it captures convexities (concavities) on the shape boundary. It is robust to boundary noise and irregularities. It is compact and it has low computational complexity.

For open curved boundaries, CSSD is not suitable because circular shifting for rotation invariance and CSSD normalization for scale invariance is not possible. For these reasons, CSSD is not suitable for coastline matching.

3.3. Fourier Descriptors

One of the most widely used shape descriptors in shape recognition is Fourier descriptor (FD). It is used in applications such as image retrieval (Zhang and Lu 2002), medical imaging, (Liang et al 1994 and Rangayyan et al 1997), hand pose recognition (Harding and Ellis 2004). Fourier descriptors are usually derived from spectral transform on the shape signatures. Global shape features are captured by the first few low frequency terms, while finer features of the shape are captured by higher frequency terms.

Many types of FD methods are used in the shape recognition literature (Zhang and Lu 2005). In these methods, different shape signatures have been employed to

extract FD. In this section FDs which are derived from shape signatures such as complex coordinates, centroid distance, curvature, and cumulative angular function will be discussed.

3.3.1. Complex Coordinates

The boundary of a shape can be represented as a sequence of coordinates $C(k) = [x(k), y(k)]$, for k= 0, 1, 2…N-1 where N is the number of boundary pixels. Each coordinate pair can be considered as a complex number such that:

$$C(k) = x(k) + jy(k). \tag{3.6}$$

This representation reduces a 2-D problem to a 1-D problem. The discrete Fourier transform (DFT) of C(k) is

$$FD(u) = \frac{1}{N}\sum_{k=0}^{N-1} C(k)\, e^{(-j2\pi uk/N)} \quad \text{for } u = 0,1,2,...,N-1. \tag{3.7}$$

The complex coefficients FD(u) are called the complex coordinates Fourier descriptors (CCFD).

3.3.2. Centroid Distance

Centroid distance is expressed by the distance of the boundary pixels to the centroid.

$$r(k) = \{[x(k) - x_c]^2 + [y(k) - y_c]^2\} \quad \text{for } k = 0,1,2,...,N-1. \tag{3.8}$$

where (x_c, y_c) is the centroid of the shape. Centroid distance function represents the deviation of the shape from a circle having a radius which equals to the mean of the centroid distance function. Figure 3.3 shows the centroid distance function of three apples.

Figure 3.3. Three apple shapes on the top and their respective centroid distance signatures at the bottom. (Source: (Zhang and Lu 2005))

Centroid distance function also reduces a 2-D problem to a 1-D problem. The discrete Fourier transform (DFT) of r(k) is

$$FD(u) = \frac{1}{N}\sum_{k=0}^{N-1} r(k)\, e^{(-j2\pi uk/N)} \quad \text{for } u = 0,1,2,...,N-1. \tag{3.9}$$

The coefficients FD(u) are called the centroid distance Fourier descriptors (CDFD).

3.3.3. Cumulative Angular Function

The change of angular directions is important to human perception. With this insight shape can be represented by its boundary tangent angles:

$$\theta(k) = \arctan \frac{y(k) - y(k-w)}{x(k) - x(k-w)} \tag{3.10}$$

where an integer w is a jump step used in practice. However, the tangent angle function $\theta(k)$ can only assume values in a range of length 2π, in the interval of $[-\pi, \pi]$ or $[0, 2\pi]$. Therefore, $\theta(k)$ in general contains discontinuities at intervals of 2π. Because of this, a cumulative angular function is introduced to overcome this problem. The cumulative angular function $\varphi(k)$ is the net amount of angular bend between the starting position C(0) and position C(k) on the shape boundary:

$$\varphi(k) = [\theta(k) - \theta(0)] \bmod(2\pi) \qquad k \in [0, L] \tag{3.11}$$

where L is the shape perimeter. The normalized variant of $\psi(t)$ is defined by Zahn and Roskies [Zahn and Roskies 72] using normalized arclength (assuming boundary is traced counter clock-wise).

$$\psi(t) = \varphi(\frac{L}{2\pi} t) - t \quad \text{where } t = \frac{2\pi k}{L} . \tag{3.12}$$

$\psi(t)$ is invariant under translation, rotation and scaling. Figure 3.4 shows the cumulative angle function, $\psi(t)$, of three apples.

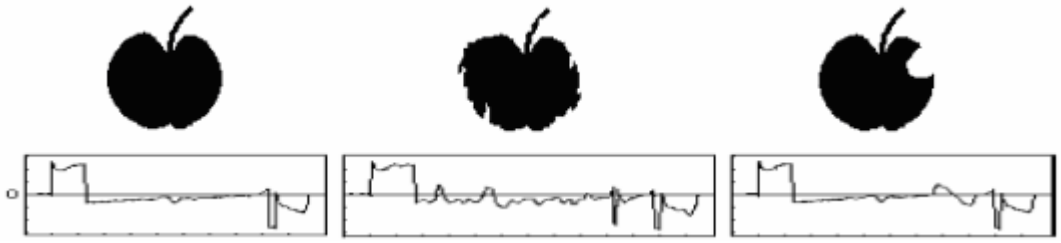


Figure 3.4. Three apple shapes on the top and their respective $\psi(t)$ at the bottom. (Source: (Zhang and Lu 2005))

The cumulative angle function also reduces the 2-D problem to a 1-D problem. The discrete Fourier transform (DFT) of ψ(k) is

$$FD(u) = \frac{1}{N}\sum_{k=0}^{N-1} \psi(k)\, e^{(-j2\pi uk/N)} \quad \text{for } u = 0,1,2,...,N-1. \tag{3.13}$$

The coefficients FD(u) are called the cumulative angular function Fourier descriptors (CAFD).

3.3.4. Curvature Signature

Curvature represents the second derivative of the boundary and the first derivative of the boundary tangent. Curvature function is given by

$$\kappa(k) := \theta(k) - \theta(k-1) \tag{3.14}$$

where is θ defined in (3.10). But this curvature function has discontinuities at size of 2π in the boundary, it is convenient to use curvature function given as:

$$\kappa(k) := \varphi(k) - \varphi(k-1) \tag{3.15}$$

where is φ defined in (3.11).Curvature is invariant under translation and rotation. The cumulative angular function reduces a 2-D problem to a 1-D problem. The discrete Fourier transform (DFT) of κ(k) is

$$FD(u) = \frac{1}{N}\sum_{k=0}^{N-1} \kappa(k)\, e^{(-j2\pi uk/N)} \quad \text{for } u = 0,1,2,...,N-1. \tag{3.16}$$

The coefficients FD(u) are called the Curvature Fourier descriptors (CFD).

3.3.5. Invariance of Fourier Descriptors to Transformations

In order to achieve shape invariance, Fourier descriptors exploit invariant properties of shape signatures and properties of Fourier transform shown in Table 3.1. Three of shape signatures discussed so far (CDFDs, CAFDs, CFDs) are translation invariant. Rotation invariance of FDs is achieved by taking only magnitude values of the FDs.

Table 3.1 Some Basic Properties of Fourier Descriptors (Source: (Gonzalez and Woods 1992))

Transformation	Boundary	Fourier Descriptor
Identity	$C(k)$	$FD(u)$
Rotation	$C_r(k)=C(k)e^{j\theta}$	$FD_r(u)= FD(u)\ e^{j\theta}$
Translation	$C_t(k)=C(k)+\Delta_{xy}$	$FD_t(u)= FD(u)+\Delta_{xy}\,\delta(u)$
Scaling	$C_s(k)=\alpha\ C(k)$	$FD_s(u)= \alpha\ FD(u)$
Starting Point	$C_p(k)=C(k-p)$	$FD_r(u)= FD(u)\ e^{j2\pi up/N}$

For complex coordinates signature, all of the components except for the first (DC) component are necessary for the invariant feature vector of the shape, since the DC component only gives the information on the position of the shape which is not useful in describing shape. Thus, translation invariance is achieved, because translation affects only first component as shown in Table 3.1. Scale and rotation invariance is achieved by dividing the magnitude of all components by the magnitude of the second component. The invariant feature vector for complex coordinates signature is then given by:

$$f = [\frac{|FD_2|}{|FD_1|}, \frac{|FD_3|}{|FD_1|}, ..., \frac{|FD_{N-1}|}{|FD_1|}]. \quad (3.17)$$

For centroid distance and curvature signatures, only half of the descriptors are necessary for the invariant feature vector of the shape, since the functions of them are real-valued and then there are only N/2 different frequencies in the Fourier transform. Scale invariance is achieved by dividing the magnitude of first half components by the magnitude of the DC component.

The invariant feature vector for centroid distance and curvature signatures is then given by:

$$f = [\frac{|FD_1|}{|FD_0|}, \frac{|FD_2|}{|FD_0|}, ..., \frac{|FD_{N/2}|}{|FD_0|}]. \quad (3.18)$$

Cumulative angular function is itself translation, rotation and scale invariant. Due to its real value, only first half of the components is necessary for the invariant feature vector of the shape. The feature vector for cumulative angular function is then given by:

$$f = [|FD_0|, |FD_1|, ..., |FD_{N/2}|]. \quad (3.19)$$

All of the four FD descriptors (complex coordinates, centroid distance, curvature and cumulative angular function) are starting point invariant, because the magnitude of Fourier transform is used as FD descriptor for all of the four signatures.

In order to compute cost of matching a model shape having a feature vector $f_m = [|f_{m1}|, |f_{m2}|, \ldots, |f_{mNC}|]$ and a query shape having a feature vector $f_q = [|f_{q1}|, |f_{q2}|, \ldots, |f_{qNC}|]$, Euclidean distance between the two feature vectors can be used as similarity measurement:

$$\text{Cost} = \left(\sum_{i=0}^{NC} \left| f_{mi} - f_{qi} \right|^2\right)^{1/2} \tag{3.20}$$

where NC is the reduced number of Fourier components needed for the feature vector. The model shape giving minimum matching cost is selected as the best match.

3.3.6. Properties of Fourier Descriptors

For closed boundaries, FDs have properties similar to CSSD. They are usually meaningful and they capture structural features of the shape boundary. They are robust to boundary noise and irregularities. They can be calculated with low computational complexity.

For open curved boundaries, FDs are not suitable because while they it capture structural features of the shape boundary, they need whole boundary information. During this process spatial locations of the object boundary are lost. For these reasons, FDs are not suitable for coastline matching.

3.4. Discussion

Extracted coastlines from aerial coastline images usually match a part of the coastlines in the map data. This means that most of the extracted coastlines are open curves. For this reason, a coastline matching method must be suitable for open curve matching. Also, a coastline matching method must be invariant to geometric transformations and must be robust to noise and deformations.

Chain code representation is sensitive to noise and deformations and it is not suitable for open curve matching. Although FD and CSSD are invariant and robust to noise, they are not suitable for open curve matching.

Shape matching and retrieval using dynamic programming is a shape matching method which was reported in (Petrakis et al. 2002). This method is robust for the case, where an open curve matches the whole or only a part of another open or closed curve,

which mostly takes place in coastline matching. It is invariant to geometric transformations and robust to noise and deformations. For this reason, we developed a coastline matching procedure using dynamic programming. Details of our approach are given in Chapter 4.

CHAPTER 4

AUTOMATIC MATCHING OF COASTLINES USING DYNAMIC PROGRAMMING

The coastline matching algorithm using dynamic programming is adapted from Petrakis, Diplaros and Millios' algorithm (Petrakis et al. 2002) which was originally described for object recognition. The algorithm is applicable to distorted and noisy coastlines by allowing matching of merged sequences consecutive segments in a coastline with the segments of another coastline. The algorithm is also invariant to translation, scale, rotation and starting point selection. The main idea of this methodology is to represent each coastline by a sequence of concave and convex segments and allow the matching of merged sequences of small segments in a deformed or noisy coastline with larger segments in the other coastline. Merging shows similar effect to that of smoothing several small segments in a coastline to form a single larger segment without performing the costly smoothing operation. The algorithm selects most promising merges based on local information. The algorithm can handle occlusion if occlusion boundaries such as boundary of clouds are identified.

4.1. Methodology

The coastline matching algorithm takes in two coastlines, determines whether coastlines are open or closed, and calculates:

1. The distance between the coastlines (dissimilarity); the more similar the coastlines are the lower the value of dissimilarity cost.
2. The correspondences between similar parts of the two coastlines.

In matching of two coastlines, A and B, the algorithm builds a dynamic Programming (DP) table (Figure 4.1), where rows and columns correspond to inflection points of A and B, respectively. By starting at any cell at the bottom row and going upwards and to the right, the table is filled with the cost of the partial match including the segments between the inflection points (rows and columns) passed so far. Only

about half of the cells are assigned cost values, in DP table, as convex segments cannot match concave ones. Merges can take place when a segment sequence of one coastline matches a single segment or a group of segments of the other coastline. Merges produce "jumps" in the traversal of the DP table. Reaching the top row (termination area) implies a complete match, when all inflection points of coastline A have been passed. Additional information is stored in each cell to allow the backtracking of a path. The backtracking of a path shows segment associations between the two coastlines. Dynamic Programming is used to find the minimum cost path from a cell in the initialization area to one in the termination area.

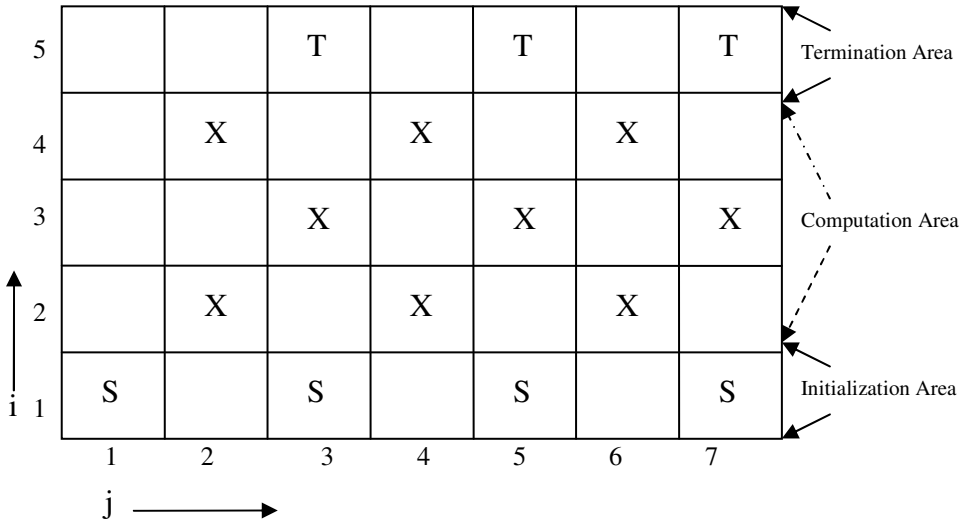


Figure 4.1. Example of a DP table with R=5 (coastline A) and L=7 (coastline B). S, X, and T represent cells in the initialization, computation, and termination areas, respectively. (Source: (Petrakis et al. 2002))

4.1.1. Segment Representation of Coastlines

In the first chapter of this thesis, we extracted the coastlines as parametric curves. Here, we use the parametric curve representation to extract dominant and robust features. First, coastlines are partitioned into segments from which visually meaningful features can be extracted. Concave (V) and convex (C) segments of a parametric curve (coastline) which are the parts of the curve between the consecutive inflection points can describe visually prominent parts of the curve. In order to extract the segments of

the curve, all inflection points of the curve are detected by computing curvature of points. Curvature of the curve can be computed from (3.1), (3.2), and (3.3) as:

$$\kappa(u,\sigma) = \frac{\dot{X}(u,\sigma).\ddot{Y}(u,\sigma) - \ddot{X}(u,\sigma).\dot{Y}(u,\sigma)}{\left(\dot{X}^2(u,\sigma) + \dot{Y}^2(u,\sigma)\right)^{3/2}}. \qquad (4.1)$$

As the parameter σ increases, the shape of curve, C_σ is smoothed due to gaussian smoothing. Curve smoothing prior to curvature computation reduces the effect of noise. After computation of curvature values for all curve points, points which have neighbors that have curvature values with opposite sign are firstly selected as the candidates of inflection points. For the next step, one of the pairs of candidate neighbor points which has low absolute curvature value is selected as the inflection point, since zero curvature produces two neighbor zero-crossing points. In this thesis, smoothing parameter σ is selected as σ=4, because experiments showed that σ=4 produced the most stable inflection points, removing the noise on the curve effectively.

After finding inflection points of the coastlines (curves), the coastlines are segmented into concave and convex segments according to the signs of the segment curvatures. The part of the coastlines between the consecutive inflection points are selected as coastline segments.

Let A and B be the two coastlines to be matched. Segments of A and B are indexed by i and j, respectively; inflection points are represented by p_i and p_j. A= a_1, a_2,……, a_M and B=b_1, b_2,…, b_N denote the sequences of M and N concave and convex segments of the two coastlines. a_i is the segment between inflection points p_i and p_{i+1} and b_j is the segment between inflection points q_j and q_{j+1}. a(i-m | i), m≥0, denotes the sequences of segments a_{i-m}, a_{i-m+1}… a_i ; similarly b(j-n | j), n≥0 are defined. If coastline (A or B) is closed, then $p_1 = p_{M+1}$ (or $q_1 = q_{N+1}$) since the number of inflection points in closed curves equals the number of segments. If coastline (A or B) is open, then $p_1 \neq p_{M+1}$ (or $q_1 \neq q_{N+1}$) the number of inflection points equals M+1 (or N+1).

4.1.2. Matching Types and Cases

Let A and B be the two coastlines to be matched. The two types of matching can be defined as:

1. **Global.** The algorithm finds the best mapping between segments in A and segments in B so that all segments are matched in each coastline.
2. **Local.** The algorithm finds the best mapping association of all segments of A to all or to a partial sequence of segments of B (i.e. some parts of B may remain unmatched) or vice versa. Calculating a suitable scale for matching and looking for which part of a coastline matches the other coastline are the key issues for this type of matching.

Coastlines A and B can be either open or closed. Depending on types of coastlines, the following matching cases should be taken into consideration:

1. **Both coastlines are open.** Matching type is local. It cannot be known in advance which coastline included within the other one, the algorithm should run twice (once for each possibility) and the matching with the minimum cost should be taken. Local matching also considers the case where all segments from both coastlines are matched (global matching).
2. **Coastline A is open and coastline B is closed.** Matching is local. Coastline A may be contained in Coastline B, but not true for reverse case (part of B may be left unmatched). Again, this case contains the case where all segments of coastline A matches all segments of coastline B (global matching).
3. **Both coastlines are closed.** Matching type is global. This case reduces to the previous case by assuming that A is open and B is closed, repeating the algorithm for global open and closed coastline matching for each possible starting point of coastline A, and by choosing the least cost match as the cost of matching.

4.1.3. Dynamic Programming (DP) Table

The algorithm builds up a dynamic programming (DP) table of partial matches and a matching between A and B is searched in the form of a path in the DP table that minimizes the total cost. The DP table has R rows and L columns, where R and L are depend on the number of segments of the two coastlines as follows.

1. **Both coastlines are open:** R=M+1 and L=N+1.
2. **Coastline A is open and coastline B is closed.** R=M+1 and L=2N. Coastline B is repeated twice to force the algorithm to consider all starting points on B. If A is closed and B is open, the roles of A and B are exchanged.
3. **Both coastlines are closed.** This case is same as the previous case.

The rows of a DP table are indexed by i, $1 \leq i \leq R$ and its columns are indexed by j, $1 \leq j \leq L$ where, i, j are indices to inflection points of A and B respectively. If coastline B is closed, its indices are found by modulo N.

The cell at the position of row i and column j is denoted by cell (i,j). A link between cells (i_{w-1},j_{w-1}) and (i_w,j_w) denotes the matching of the merged sequence of segments $a(i_{w-1}+1 \mid i_w)$ with $b(j_{w-1}+1 \mid j_w)$. A path is a linked sequence of cells $((i_0,j_0), (i_1,j_1), \ldots (i_t,j_t))$, not necessarily adjacent , indicating a partial match, where $i_0 \leq i_1 \leq \ldots \leq i_t$ and $j_0 \leq j_1 \leq \ldots \leq j_t$. This path begins at inflection point p_{i0} of coastline A and at inflection point q_{j0} of coastline B and tries to match sequences of segments $a(i_{w-1}+1 \mid i_w)$ of A with the sequences $b(j_{w-1}+1 \mid j_w)$ of B for w = 1,2,…,t. The previous cell of $cell(i_w,j_w)$ is represented by $cell(i_{w-1},j_{w-1})$ and is called parent of $cell(i_w,j_w)$.

Each $cell(i_w,j_w)$ contains the following values: $g(i_w,j_w)$, u_w, v_w, m_w, n_w,and ρ_w where $g(i_w,j_w)$ is the partially accumulated match coast up to the current cell, u_w and v_w denote the number of unmatched segments of A and B respectively, m_w and n_w are the indices of the parent cell of $cell(i_w,j_w)$ which means , $m_w = i_{w-1}$ and $n_w = j_{w-1}$ and are used to backtrack a path. ρ_w denotes the scale factor corresponding to the parts of A and B which have been matched up to $cell(i_w,j_w)$. Figure 4.2 shows an example of a DP table. The DP table consists of following three areas:

1. Initialization area: The first row of the DP table is initialization area. All paths start from the cells in this area. Matching begins always from

the first segment a_1 of A (i_0=1). Matching may start with any segment b_{j0} of B, where $1 \leq j_0 \leq L$. If a_1 and b_{j0} have same curvature polarity, then, $g(1,j_0)$, m_w, n_w, u_w, v_w, and ρ_w are0, 0, 0, R, L, 1, respectively;
Otherwise $g(1,j_0) = \infty$.

2. Computation area. The area between the first and last row of the DP table is computation area. Cells in this area correspond to incomplete paths.

3. Termination area. The last row of the DP table is termination area. All complete paths end at the cells in this area. The best match corresponds to the cell with the least cost.

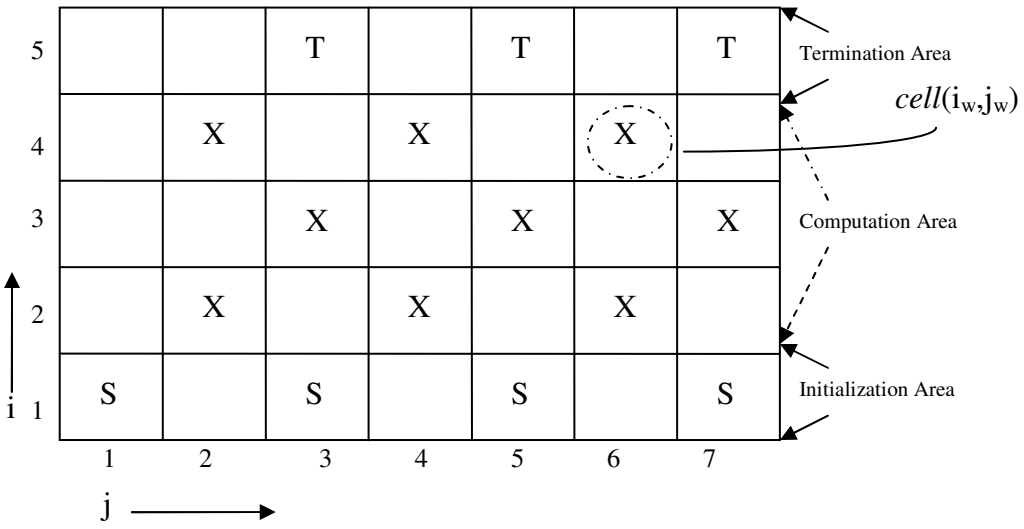


Figure 4.2. Example of a DP table with R=5 (coastline A) and L=7 (coastline B). S, C, and V represent cells in the initialization, computation, and complete match areas, respectively. (Source: (Petrakis et al. 2002))

About half of the cells of DP table are empty, because matching between opposite type segments (i.e., C and V) are not allowed. So, the cost of matching is set to infinitive to avoid matching of opposite type segments. The first column of DP table is also empty except the cell in the first row, since matching whole of coastline A with whole or a part of coastline B is allowed but reverse condition is not allowed. Matching always begins from the first inflection point of A while any point of B is a candidate starting point. Figure 4.2 implies that the first segments of A and B have the same

polarity; otherwise matching starts from the second segment of B.

4.1.4. Cost Function

A complete match is a correspondence between the sequences of segments in order, so that no segments are left unmatched in coastline A and there are no crossovers. A complete match is characterized by a complete path ((i_0,j_0), (i_1,j_1), ...,(i_T,j_T)), which a path that starts from the initialization area and ends at the termination area. The cost (distance) D(A,B) of matching coastline A with coastline B can be defined as;

$$D(A,B) = \min_{T}\{g(i_T, j_T)\} \tag{4.2}$$

where $g(i_T,j_T)$ is the cost of the complete match. $g(i_T,j_T)$ is defined by:

$$g(i_T, j_T) = \min_{i_w, j_w} \sum_{w=1}^{T} \psi(a(i_{w-1}+1 \,|\, i_w), b(j_{w-1}+1 \,|\, j_w). \tag{4.3}$$

Function $\psi(a(i_{w-1}+1 \mid i_w),\ b(j_{w-1}+1 \mid j_w))$ represents the similarity cost of two arguments and contains three additive components:

$$\begin{aligned}\psi(a(i_{w-1}+1|i_w), b(j_{w-1}+1|j_w)) = \lambda.(&M_C(a(i_{w-1}+1|i_w)) + M_C(b(j_{w-1}+1|j_w)) \\ &+ D_C(a(i_{w-1}+1|i_w), b(j_{w-1}+1|j_w))\end{aligned} \tag{4.4}$$

where M_C is merging cost and D_C is dissimilarity cost. The first two term in (4.4) represent the cost of merging segments $a(i_{w-1}+1 \mid i_w)$ in coastline A and segments $b(j_{w-1}+1 \mid j_w)$ in coastline B respectively while last term is the cost of matching the merged sequence $a(i_{w-1}+1 \mid i_w)$ with the merged sequence $b(j_{w-1}+1 \mid j_w)$. Constant λ

denotes the relative importance of the merging costs. Low values of λ encourage merging and high values of λ inhibit merging. Low values of λ should be used while matching of the coastlines with high amount of detail.

Merging should the follow the grammar rules such that each allowable merging should be a recursive application of the grammar rules CVC→C and VCV→V.

4.1.5. Segment Features

Merging a "visually prominent" segment (a large segment with high curvature) into merged segment of opposite type (convex or concave) should produce higher cost. To specify this cost requirement, "visually prominence" should be defined as the segment features in geometric terms.

The partial cost components calculated from different features of the coastline segments should be associated to a total cost in a visually meaningful way.

In the specification of visual prominence of a segment, three segment features shown in Figure 4.3 are defined as:

1. **Rotation Angle** θ_i is the angle traversed by the tangent to the segment from first inflection point p_i of the segment a_i to the other inflection point p_{i+1} of the segment a_i . Rotation angle shows how strongly a segment is curved.
2. **Length** l_i is the length of the segment a_i.
3. **Area** A_i is the area enclosed by the chord and the line between the inflection point p_i and p_{i+1} of the segment a_i.

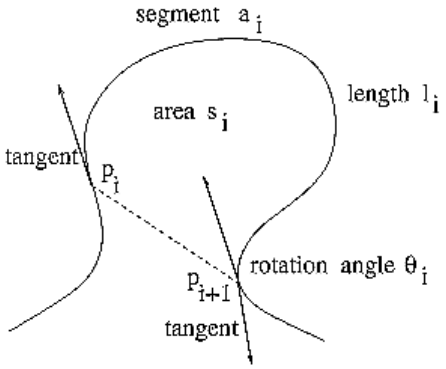


Figure 4.3. Segment features for defining the importance of a segment. (Source: (Petrakis et al. 2002))

4.1.6. Scale Factor

The length of one of two coastlines has to be multiplied by an appropriate **scale factor**, in case one of two coastlines with one scaled with respect to the other. Scale factor can be computed as the ratio of the lengths of the matched parts of coastlines A and B respectively. Scale factor is computed due to matching types.

Global matching: Coastline A matches all segments of coastline B. In other words, the algorithm matches all segments from both coastlines. The scale factor is a constant and is calculated by,

$$\rho = \frac{l(A)}{l(B)} \tag{4.5}$$

where *l(A)* and *l(B)* are the lengths of coastline A and B respectively

Local Matching: Coastline A may match either the whole or a part of coastline B. This case is the generalized form of the global matching case. The length of the matched part of coastline B is unknown until the DP table is completely filled, although the matched part of coastline A is the whole coastline A. To solve this problem, a scale factor ρ_t is computed for each partial path $((i_0,j_0), (i_1,j_1)...(i_t,j_t))$, corresponding to the matched parts:

$$\rho_t = \frac{\sum_{w=1}^{t-1} \sum_{i=i_{w-1}}^{i_w-1} l_i(A)}{\sum_{w=1}^{t-1} \sum_{j=j_{w-1}}^{j_w-1} l_j(B)} \tag{4.6}$$

where $2 \leq t \leq T$ and $l_i(A)$ and $l_j(B)$ are the lengths of a_i and b_j, respectively. This value is an approximation of the actual scale factor of a complete match. ρ_1 is not included in computation, since the total matched length is 0 for two coastlines. It is convenient to set ρ_1 to 1.

4.1.7. Dissimilarity Cost

The dissimilarity cost of matching a group of the segments from coastline A with a group of the segments from coastline B is computed by:

$$D_C = U.\max_f \{|d_f|\}. \tag{4.7}$$

The intuition behind the use of maximum argument is to emphasize large differences on any feature. The term d_f is the cost associated with the distance in feature f (i.e., length, area or angle). d_f may be negative when f is angle.

$$d_f = \frac{|F_A - S_w(f)F_B|}{F_A + S_w(f)F_B} \tag{4.8}$$

where, $F_A = \sum_{i=i_{w-1}+1}^{i_w} |f_i|$, $F_B = \sum_{j=j_{w-1}+1}^{j_w} |f_j|$ and $S_w(f)$ is a parameter depending on the feature f. $S_w(f)=\rho_{w-1}$ for f being length and $(\rho_{w-1})^2$ for f being area. For f being rotation angle, $S_w(f) =1$ because angle does not depend on the scale. ρ_{w-1} is computed according to (4.5) and (4.6) for global and local matching respectively.

U is a weight term associated with the dissimilarity of the partial match such that: U emphasizes the importance of matching large parts from both coastlines, in a similar way that human pay more attention on large coastline parts when judging the quality of matching. Without U, the matching of very small coastline parts contributes to the cost of matching equally with the matching of large parts. U is defined as the maximum of two proportions of the matched coastline length with respect to the total length:

$$U := max\left\{ \frac{\sum_{i=i_{w-1}+1}^{i_w} l_i(A)}{l_A}, \frac{\sum_{j=j_{w-1}+1}^{j_w} l_j(B)}{l_B} \right\}. \quad (4.9)$$

4.1.8. Merging Cost

Let the types of the segments being merged be CVC → C, Segments are merged to a single merged convex segment C by absorbing the concave segments between convex segments. The opposite case is obtained by exchanging C and V in the formulas. The merging cost can be defined by:

$$M_C = \max_f \{U_f C_f\} \quad (4.10)$$

where f represents a feature (length, area or rotation angle).

For all features:

$$C_f = \frac{\sum_{\text{V segs of group}} |f|}{\sum_{\text{all segs of group}} |f|} \quad (4.11)$$

where the sum in the numerator is for the absorbed concave segments, while the sum in the denominator is for all merged segments of the group. The intuition behind this formula is to measure the importance of the absorbed segments relative to all merged segments of the group.

The weight term of merging cost is defined as:

$$U_f = \frac{\sum_{\text{V segs of group}} |f|}{\sum_{\text{V segs of coastline}} |f|} \tag{4.12}$$

where the sum in the numerator is for the absorbed concave segments, while the sum in the denominator is for all concave segments of the coastline. The intuition behind weight term is to measure the importance of the absorbed segments within the whole coastline.

4.2. DP Algorithm for Matching Coastlines

In previous section of this chapter, methodology and definitions of DP algorithm have been discussed. In this section, The DP algorithm will be outlined and issues about optimality and complexity will be discussed.

4.2.1. Outline of DP Algorithm

Let A and B be the two coastlines to be matched. A is assumed to be open; B may be either open or closed. If both coastlines are closed, A is assumed to be open and it is attempted to match the open A on closed B. Each point of A is a candidate starting point for matching. Matching starts with segments which have same polarity (C or V). There are M/2 such segments (potential starting points) on A and the algorithm is repeated M/2 times where M is the number of the segments of coastline A. It's assumed that the first segment of B has the same polarity with the first segment of A; otherwise, matching starts with the second segment of B. The last matched segments of A and B must also have same polarity.

A summary about the above discussion according to Figure 4.2 follows as:

1. Global matching. The algorithm consumes all segments from both coastlines. The algorithm starts at the left-most cell (marked as "S", i.e. initialization area) of the DP table, proceeds upwards and to right through cells of computation area

(marked as “X”), and terminates at the at the right-most cell (marked as “T”, i.e. termination area) of the DP table corresponding to same polarity segments of A and B. This cell contains the overall cost of matching. The scale factor is computed according to (4.5).

2. Local matching. Any segment on coastline B is a candidate starting segment for matching, if it has the same polarity with the first segment of coastline A. Only half of cells (marked as “S”) in the initialization area are candidate cells for starting of a path. The algorithm consumes some or all segments of B and may end at any segment of B having the same polarity with the last segment of A. Half of the cells (marked as “T”) in the termination area are candidate termination cells of a complete match path. All candidate termination cells should be searched for the least cost match (best match). The scale factor is computed due to (4.6).

Outline of the matching algorithm is shown in Figure 4.4. The algorithm computes the cost D(A,B) of two input coastlines. The *for* loop for j_w does not run for all values of j_w, since convex to concave matches are not allowed. At each cell, the algorithm computes the optimum cost of incomplete path ending at this cell. That is

$$g(i_w, j_w) = \min\{g(i_{w-1}, j_{w-1}) + \psi(a(i_{w-1}+1 \,|\, i_w), b(j_{w-1}+1 \,|\, j_w)\}. \tag{4.13}$$

Equation (4.13) determines the minimum cost transition from cell (i_{w-1}, j_{w-1}) to (i_w, j_w) for all possible values of i_{w-1} and j_{w-1}. Merging always contains an odd number of segments;

$$(i_{w-1}, j_{w-1}) = (i_w - 2m_W - 1, j_w - 2n_W - 1) \tag{4.14}$$

where $m_w \geq 0$ and $n_w \geq 0$. Then Equation (4.13) can be written as:

$$g(i_w, j_w) = \min\{g(i_w - 2m_w - 1, j_w - 2n_w - 1) + \psi(a(i_w - 2m_w \,|\, i_w), b(j_w - 2n_w \,|\, j_w)\} \quad (4.15)$$

where $0 \le m_w \le (i_w\text{-}1)/2$ and $0 \le n_w \le (j_w\text{-}1)/2$. Indices (m_w, n_w) characterize this transition and stored at cell (i_w, j_w) and can be used to backtrack the path from cell (i_w, j_w) back to its starting point.

Input: Coastlines A=a_1, a_2,...,a_R, B=b_1, b_2,...,b_L:
Output: Cost D(A,B) and correspondences between segments.
// Initialization: Fill the first row
for j_0=1 ,2, , L **do**
if a_1 and b_{jo} are both C or V then *cell(1, j_0)=(0, 0, 0, M, N, 1);*
otherwise *cell(1, j_0)=(∞, 0, 0, M, N, 1)*;
end for

// Fill from the second to the R-th row
for i_w=2, 3,…, R **do**
for j_w=2, 3,…, L **do**
if a_1 and b_{jo} are both C or V then fill *cell(i_w, j_w)* using (4.13);
compute ρ_w using (4.5) or (4.6);
end for

// Select the least cost complete path
select the least cost path from the R-th row;
backtrack path using m_w and n_w cell values;

Figure 4.4. Outline of the DP algorithm. (Source: (Petrakis et al. 2002))

Matching always starts at the first inflection point of A (i_0=1) while any point of B is a candidate starting point. DP table is initialized by filling its first row: When a_1 and b_j have same polarities, then g(1,j), m_1, n_1, u_1, v_1 are 0, 0, 0, M, N respectively, implying that each of these cells can be a starting point. If a_1 and b_j have different

polarity, g(1,j) is set to be infinite since the matching of opposite type segments (C or V) is not allowed. If B is closed, then for $N<j \leq L$, g(1,j) is set to be infinite since cell indices j are modulo N (i.e., $b_j=b_{j-N}$) and a starting point is not needed to be calculated twice.

Equation (4.13) implies that the algorithm computes the minimum cost transition from each allowable cell (i_{w-1}, j_{w-1}) to cell (i_w, j_w). The algorithm may become very slow especially on large DP tables. Transitions on the DP table correspond to merges of segments. The algorithm examines all merges, even the less rational ones, such as merges involving all segments. It is reasonable to restrict the maximum number of segments which are allowed to merge by a constant K (i.e., $i_w-i_{w-1}=K$ or $j_w-j_{w-1}=K$) depending on the type of coastline matching problem. For global matching, $K \geq M / N$ where $M \geq N$. The algorithm may miss the least cost match if matched coastlines actually contain merging of more than K segments.

4.2.2. Invariance to symmetric coastlines and different starting points

The matching algorithm must be capable of handling symmetric shapes and alternative starting point cases. Figure 4.5 shows all these cases for an open curve A: A_1 is the original curve, A_2 is its mirror image, A_3 corresponds to opposite curve traversal of A_1 (i.e., selection of the starting point), and A_4 corresponds to the combination of A_2 and A_3. A complete representation of A consists of the representations of A_1, A_2, A_3,

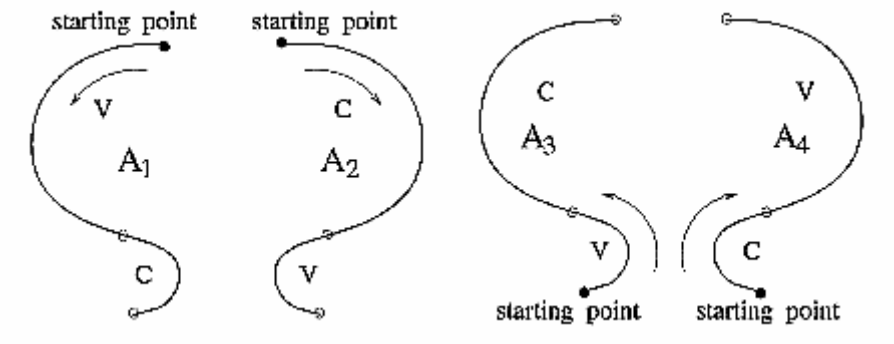


Figure 4.5. Curve representation cases. $A_1$is the original curve, A_2 is its mirror image, A_3 shows curve traversal in the opposite direction and A_4 is the mirror image of A_3 with the opposite traversal (Source: (Petrakis et al. 2002))

4.2.3. Optimality of DP Algorithm

In this section, it is proved that the coastline matching algorithm is optimal, in other words it always finds the path with the least cost. The proof (Rao et al. 1999) uses the DP table of Figure 4.6 which shows two alternative complete match paths.

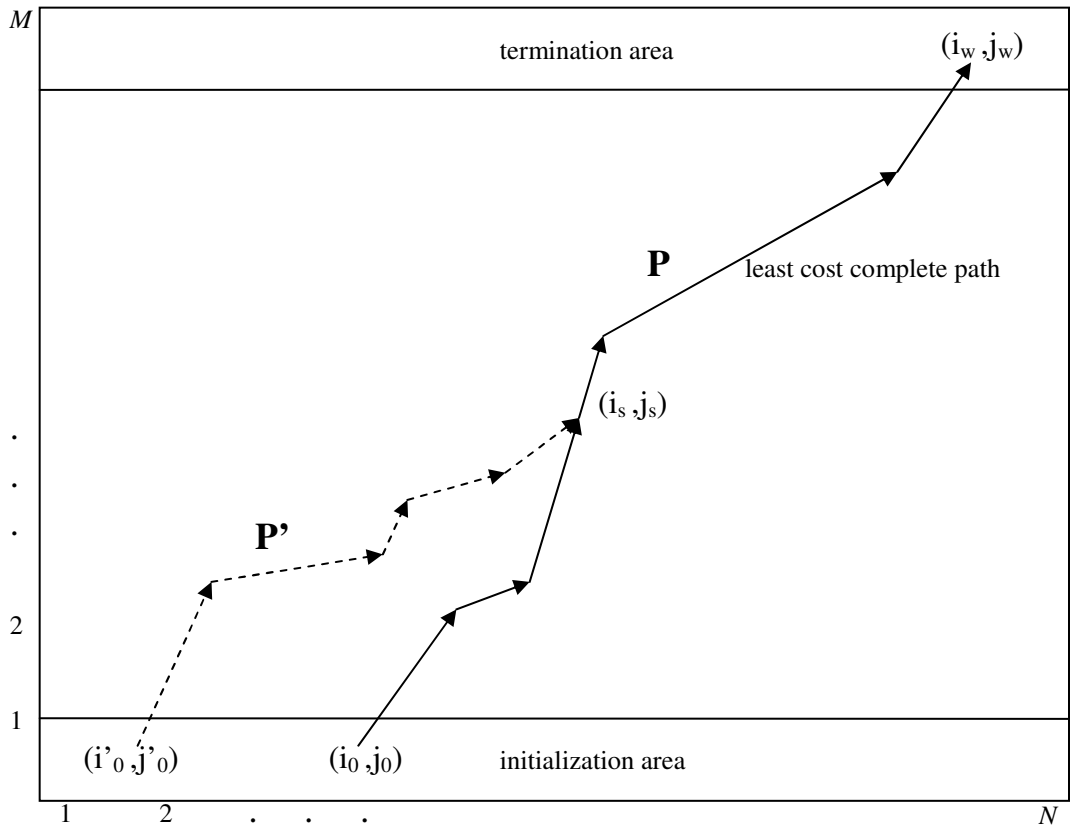


Figure 4.6. Example of DP table showing two alternative complete match paths.

Lemma 1 *The DP algorithm described in section 4.2 is optimal.*

Proof: Let P=$((i_0,j_0), (i_1,j_1),..., (i_w,j_w))$ be the best (least cost) complete path which has been found by DP algorithm. If this path is not the least cost path then, let the other path be P'=$((i_0',j_0')... (i_w,j_w))$. The second path ends at (i_w,j_w) too, since (i_w,j_w) is always selected as the end point of the path with the least cost. The two paths should meet at some cell (i_s, j_s) either before or at (i_w,j_w); the algorithm should have selected path P'

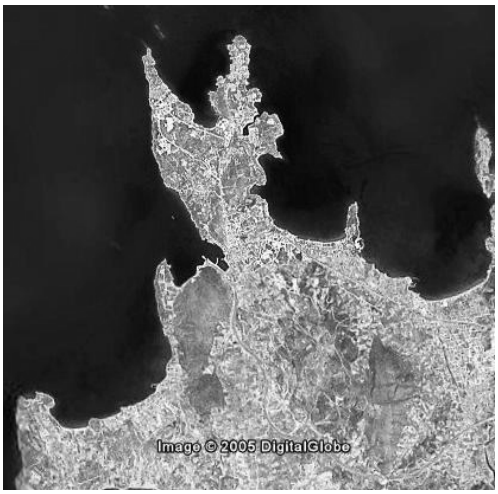
instead of path P since has less cost. If they meet at a cell (i_w,j_w), then the algorithm should have selected path P' instead of path P. If they meet at cell (i_s,j_s) before (i_w,j_w) , then the cost of P' at (i_s,j_s) should be less than the the cost of P up to that cell. But then the cost of P' at (i_w,j_w) should be less than the cost of P which leads to a contradiction.

4.2.4. Complexity of DP Algorithm

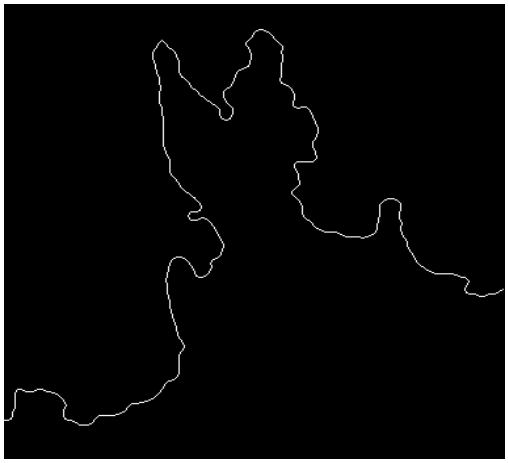
The computational complexity of the DP algorithm depends on the time of computing ψ, the cost of matching two sequences of segments. ψ is the basic operation of the DP algorithm. In Equation (4.13), the cost computation at each cell (i,j) takes i.j/2 time. The computational complexity for filling a DP table of size M x N is $O(M^2N^2)$ if at least one of the coastlines is open. If both coastlines are closed, the algorithm repeated M times so the computational complexity of the algorithm becomes $O(M^3N^2)$. By restricting merging to K segments (for K<<M,N) , the complexity becomes $O(K^2M\ N)$ for open coastlines and $O(K^2M^2\ N)$ for closed coastlines.

4.2.5. Results

In order to test the proposed coastline matching method, an 1/250000 vector map of Aegean coast of Turkey has been used. It has been taken as reference image and segmented as described in section 4.1.1. 20 Aerial coastline images which are taken at 13-14 miles altitudes have been acquired from an internet search site for aerial images (http://earth.google.com). Coastlines of the images have been extracted by using proposed coastline extraction method described in Chapter 2. The longest coastline in the coastline image has been selected to match map data and segmented as described in section 4.1.1. Segment features for the images and the map have been calculated. By proposed DP coastline matching method, 9 of 20 images have been matched with map data correctly. Some examples of correct match of coastline images and the map are shown in Figure 4.7 and in Figure 4.8.



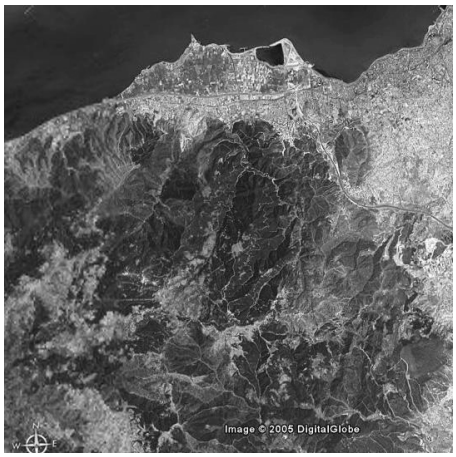
(a)



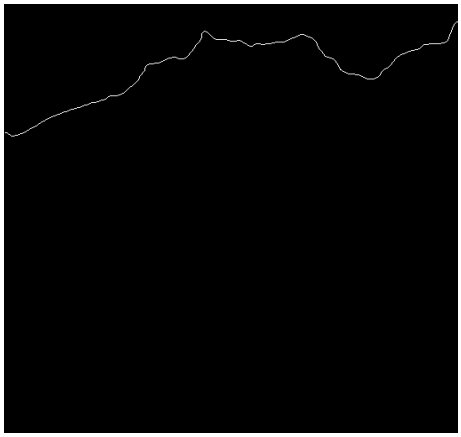
(b)

(c)

Figure 4.7. (a) Aerial image of Çeşme (b) Extracted coastline (c) Match of the coastline (coastline segment is highlighted in black).



(a)



(b)

(c)

Figure 4.8. (a) Aerial image of İnciraltı (b) Extracted coastline (c) Match of the coastline (coastline segment is highlighted in black).

CHAPTER 5

CONCLUSIONS

In this thesis, problem of matching aerial coastline images with map data has been studied. The procedure developed here is based on texture segmentation on wavelet images in order to extract coastlines and matching coastline segments with map data using dynamic programming. Some important observations can be summarized as follows:

1. Both coocurrence and histogram features can be used for discriminating sea and land textures in the wavelet image. They showed stable segmentation performance by using wavelet representation of the coastline images except very homogeneous or smooth parts of the land region and very noisy parts of the sea region.
2. Coocurrence features showed better segmentation performance for 32X32 and 16X16 pixel window resolutions while histogram features showed better segmentation performance for 8X8, 4X4 and 2X2 pixel window resolutions.
3. The performance of the maximum likelihood classifier has been tested on natural textures.
4. Proposed automatic coastline extraction method extracted coastlines from aerial coastline images with a maximum four-pixel extraction error.
5. Chain code representation, Fourier descriptors (FD) and curvature scale space descriptor (CSSD) are not suitable for coastline (open curve) matching. Shape matching using dynamic programming is suitable for open curve (coastline) and closed curve matching.
6. A shape matching method using dynamic programming has been adapted for coastline matching. 9 (45 %) of 20 aerial coastline images have been correctly matched to vector map data

The study presented here is a novel attempt to develop an automatic procedure solving an image understanding problem which may lead to many interesting applications such as cartography and autonomous aerial navigation. Although the results achieved here demonstrate how the problem can be successfully solved, the

performance of the solution can be further improved. For the future work, firstly we can consider improving the accuracy and speed of proposed the coastline extraction and the coastline matching method. By using of parallel processing algorithms, computation time of both methods may be decreased. In this thesis we focused on sea and land segmentation to extract coastline for the coastline images that does not contain clouds. For the coastline images with clouds, detection of clouds should be a primary focus of interest.

BIBLIOGRAPHY

Bijaoui, J., Cauneau, F. 1994. “Separation of sea and land in SAR images using texture classification” 'Oceans Engineering for Today's Technology and Tomorrow's Preservation.' Proceedings, Brest, (13-16 September) Vol.1, pp.522-526

Bo, G., Dellepiane, S., De Laurentiis, R. 2000 “Semiautomatic Coastline Detection in Remotely Sensed Images” IEEE International Geoscience and Remote Sensing Symposium, Honolulu, (24-28 July) Vol.5 pp. 1869-1871

Bo, G., Dellepiane, S., De Laurentiis, R. 2001 “Coastline Extraction in Remotely Sensed Images by Means of Texture Features Analysis” IEEE International Geoscience and Remote Sensing Symposium, Sydney,(9-13 July) Vol.3 pp. 1493-1495

Chin-Chen, C., I-Yen, C., Yea-Shuan H. 2002. “Hand Pose Recognition Using Curvature Scale Space” Proceedings of the 16th International Conference on Pattern Recognition. (11-15 August) Vol.2, pp. 386-389

Daubechies, I., 1992. “Ten Lectures on Wavelets” CBMS-NSF Regional Conference Series in Applied Mathematics, Vol.61 pp. 195.

Duda R.O., Hart P.E., Stork D.G., 2001. *Pattern Classification,* (John Wiley, New York) p. 85-89.

Eugenio, F., Marques, F., Marcello, J. 2002 “A Contour-based Approach to Automatic and Accurate Registration of Multitemporal and Multisensor Satellite Imagery” IEEE International Geoscience and Remote Sensing Symposium, (24-28 June) Vol.6, pp. 3390-3392

Freeman, H. 1961. “On The Encoding of Arbitrary Geometric Configurations” IRE Trans. Electron. Comput., EC-I0 No.2 (June) pp.260-268.

Gonzalez R.C. Woods R.E. 1992. *Digital Image Processing* (Addison-Wesley) p. 492-494.

Gyeonghwan K., Govindaraju, V.1997. “A Lexicon Driven Approach to Handwritten Word Recognition for Real-time Applications” .IEEE Trans Pattern Anal. Machine Intell., Vol.19, No.4, pp.366-379

Haralick, R.M., Schanmugan, K., and Dinstein, I. 1973. “Texture Features for Image Classification.” IEEE Trans. on Syst., Man, Cybern., SMC, Vol.3, No.6, pp. 610-621.

Harding, P.R.G., Ellis, T. 2004. “Recognizing Hand Gesture Using Fourier Descriptors” Proceedings of the 17th International Conference on Pattern Recognition. (23-26 August) Vol.3, pp. 286-289

Jager, F.; Koren, I.; Gyergyek, L. 1990. **"**Multiresolution Representation and Analysis of ECG Waveforms**"** Computers in Cardiology Proceedings, Chicago, (23-26 September) pp.547-550

Jeannin, S. (Ed.) 2000, "MPEG-7 Visual Part of Experimentation." Model Version 5.0. ISO/IEC JTC1/SC29/WG11/N3321, Nordwijkerhout, March.

Jianbin, Xu., Wen, Hong., Zhe, Liu., Yirong, Wu., Maosheng, Xiang., 2003. "The Study of Rough-location of Remote Sensing Image with Coastlines" IEEE International Geoscience and Remote Sensing Symposium, (21-25 July) Vol.6, pp. 3964-3966

Jishuang, Q., Chao, W., Zhengzhi, W., 2002."Coastal Line Feature Extraction Method for Optic Remote Sensing Images: A Threshold-based Morphological Approach" IEEE International Geoscience and Remote Sensing Symposium, (24-28 June) Vol.6, pp. 3420-3422

Liang, S., Rangayyan, R.M., Desautels, J.E.L.1994. "Application of Shape Analysis to Mammographic Calcifications" IEEE Trans. on Medical Imaging, Vol.13, No.2, pp. 263-274

Loos, E.A., Niemann, K.O. 2002. "Shoreline Feature Extraction from Remotely-sensed Imagery "IEEE International Geoscience and Remote Sensing Symposium (24-28 June) Vol.6, pp.3417-3419

Madhavan, R., Durrant-Whyte, H., Dissanayake, G. 2002 "Natural Landmark-based Autonomous Navigation Using Curvature Scale Space" IEEE International Conference on Robotics and Automation Proceedings. (11-15 May) Vol.4, pp.:3936-3941

Mallat, S. 1989. "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation." .IEEE Trans. on Pattern Anal. Machine Intell. Vol.11, No.7, pp. 674-693.

Mokhtarian, F., Mackworth A.K. 1992. "A Theory of Multi-scale, Curvature-based Shape Recognition for Planar Curves." IEEE Trans. on Pattern Anal. Machine Intell. Vol.14, No.8, pp. 789-805.

Mokhtarian, F., Abbasi, S., Kittler, J. 2000 "Enhancing CSS-based Shape Retrieval for Objects with Shallow Concavities." Image and Vision Computing, Vol.18, No.3, pp. 199-211.

Mokhtarian, F., Abbasi, S., Kittler, J. 1996. "Indexing an Image Database by Shape Content Using Curvature Scale Space" IEE Colloquium on Intelligent Image Databases, London, (22 May), Vol.4, pp. 1-6.

Mokhtarian, F., Abbasi, S. 2001. "Affine-similar Shape Retrieval: Application to Multiview 3-D Object Recognition" IEEE Trans. on Image Processing, Vol.10, No.1, pp. 131-139.

Mokhtarian, F., Abbasi, S., 2004. "Matching Shapes with Self-intersections: Application to Leaf Classification" IEEE Trans. on Image Processing, Vol.10, No.5, pp. 131-139.

Matusiak, S., Daoudi, M., Blu T. 1998. "Sketch-based Images Database Retrieval" Proceedings of the Fourth International Workshop on Advances in Multimedia Information Systems, Istanbul (24-26 September), pp. 185-191.

Petrakis, E.G.M., Diplaros, A., Millios, E. 2002. "Matching and Retrieval of Distorted and Occluded Shapes Using Dynamic Programming." IEEE Trans. on Pattern Anal. Machine Intell. Vol.24, No.11, pp. 1501-1516.

Rangayyan, R.M., El-Faramawy, N.M., Desautels, J.E.L., Alim, O.A. 1997. "Measures of Acutance and Shape for Classification of Breast Tumors" IEEE Trans. on Medical Imaging, Vol.16, No.6, pp.799-810.

Rao, Z., Petrakis, E.G:M., Millios, E. 1999. "Retrieval of Deformed and Occluded Shapes using Dynamic Programming." Technical Report CS-1999-06, Department of Computer Science, York University, pp. 16-17.

Thyagarajan, K.S., Nguyen, T., Persons, C.E. 1994. "A Maximum Likelihood Approach to Texture Classification Using Wavelet Transform" IEEE International Conference on Image Processing, Austin, (13-16 November). Vol.2, pp. 640-644.

Zahn, C.T., Roskies, R.Z. 1972. "Fourier Descriptors for Plane Closed Curves" IEEE Trans. on Computer, Vol.21, No.3, pp. 269–281.

Zhang, T.Y., Suen, C.Y. 1984 "A Fast Parallel Algorithm for Thinning Digital Patterns" Communications of the. ACM, Vol.27, No.3, pp. 236-239.

Zhang, Bin., Zhu, Zheng Zhong., Wu, You Shou. 1997. "Accurate Geometric Correction of NOAA AVHRR Images: The Coastline Detection Approach with Respect to Multispectral and Spatial Information" IEEE International Conference on Intelligent Processing Systems., Beijing, (28-31 October) Vol.2, pp.28-31

Zhang, D., Lu, G. 2002. "Enhanced Generic Fourier Descriptors for Object-based Image Retrieval" IEEE International Conference on Acoustics, Speech, and Signal Processing. (13-17 May) Vol.4, pp.3668-3671

Zhang, D., Lu, G. 2004. "Review of Shape Representation and Description Techniques" Pattern Recognition Vol.37, No.1, pp. 1–19

Zhang, D., Lu, G. 2005. "Study and evaluation of different Fourier methods for image retrieval" Image and Vision Computing. Vol.23, No.1, pp. 33–49

APPENDIX A

COMPUTER SOFTWARE

In this thesis, proposed coastline extraction and matching methods are implemented by using several computer programs written in C and Matlab. These computer programs and a text file, README, describing the computer programs are given in a CD. A brief description of the computer programs are given as follows:

MAIN_CSMATCH.C: It is the main source code file that accepts an aerial coastline image in PGM or PPM image format and a map database which contains the feature vectors of segments of the vector map as the inputs. It outputs segments of the map database corresponding to those of the aerial coastline image.

MAP_SEG_FEATURE.C: It is the source code file that accepts an ASCII text file which contains the coordinates of the vector map as the input. It outputs an ASCII text file which contains the feature vectors of segments of the map in a specified scale.

MAP_SEGMENT.C: It is the source code file that accepts an ASCII text file which contains the coordinates of the vector map and starting and ending segments of the matched segments as the inputs. It outputs the coordinates of matched segments in the vector map in a specified scale.

TRAINING.C: It is the source code file that accepts training image samples of sea or land textures in PGM image format as the inputs. It outputs an ASCII text file which contains the mean and the covariance matrix of the feature vector for training sea or land texture samples in a specified window size.

FEATURE_SELECTION.C: It is the source code file that accepts training image samples of sea or land textures in PGM image format as the inputs. It outputs an ASCII text file which contains the feature vectors for training sea or land texture samples in a specified window size.

FEATURE_SELECTION.M: It is the source code file that accepts the ASCII text files which contains the feature vectors for training sea and land texture samples in a specified window size as the inputs. It outputs the ranked importance values of the Haralick's CM features in the specified window size.

CSMATCHLIB.C: It is the library file that contains the C functions which are used in the implementation of feature extraction, the multiscale image segmentation, training, ML classifier and DP algorithms described in this thesis.

IMG_PRO.C: It is the library file that contains the C functions which are used in the wavelet decomposition and image processing procedures used in this thesis.

CSMATCHLIB.H: Header file of CSMATCHLIB.C.

IMG_PRO.H: Header file of IMG_PRO.C.