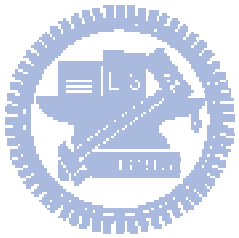
國立交通大學
資訊科學與工程研究所

博士論文

多邊形描述及其應用

Polygon Descriptor and It's Applications



研究生：賴柏伸

指導教授： 傅心家教授

中華民國九十九年十二月

多邊形描述及其應用
Polygon Descriptor and It's Applications

研 究 生： 賴柏伸　　Student: Por-Shen Lai

指導教授： 傅心家　　Advisor: Hsin-Chia Fu

國立交通大學
資訊科學與工程研究所
博士論文

A Dissertation Submitted to
Department of Computer Science
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
in Computer Science

December 2010
Hsinchu, Taiwan, R.O.C

中華民國九十九年十二月

博士論文

多邊形描述及其應用

研究生：賴柏伸　　指導教授：傅心家教授

國立交通大學 資訊科學與工程研究所

摘　要

本論文提出一個基於多邊形狀的模型，**多邊形描述模型** (Polygon description)，來將資料分佈的形狀以數值的形式來加以描述。藉由對系統因子取樣，就可以繪製資料分佈圖來顯示系統因子間的資料相依性。由於系統行為隱含在資料的相依性中，因此本文提出之用來描述資料分佈形狀的多邊形描述模型便可被用來代表抽象的系統行為。另外，為了衡量兩個多邊形描述模型間的相似度，本文提出了兩個相似度比較的方法 - 1) **變形距離估計** (Deforming distance estimation) 以及 2) **正交向量比對** (Normal vectors match)。基於一組預先定義的變形運算元 (Deforming operator)，**變形距離估計** 尋找變形所需的最小成本之運算元組合來衡量兩多邊形描述模型間的相似程度。而**正交向量比對**則透過尋找兩組正交向量間的最佳對應來對齊兩個**多邊形描述模型**，並根據正交向量間的角度差異來衡量**多邊形描述模型**間的相似度。一般來說，**變形距離估計**擁有比較良好的擴充能力但侷限於使用二維的資料。**正交向量比對**可以適用於任何在任何維度上，但要延伸應用在特定應用上卻頗為困難。

除此之外，基於多邊形描述模型，基於相似度的分群方法可以利用多邊形描述模型來加以強化。透過多邊形描述的的幫助，我們可以幫只有相似度資訊的資料建立一個虛擬的空間，並在該虛擬空間中幫每個群計算其變異度。除此之外，多邊形描述的涵蓋區域也可以用

來描述一塊區域。利用多邊形描述模型來描述的多邊形區塊將可以在各式各樣的多媒體應用中被使用，例如區域篩選，追蹤等等。透過本論文中展示的變異度強化 K-Medoid 模型可以知道多邊形的概念可以在分群方法上發揮相當的功能。

相對於提出來的三個延伸方法 1)變形估計，2)基於相似度分群法的虛擬幾何，以及 3) 多邊形區域描述，本論文展示了三個實際應用 1)財金資料挖況，2) 可變大小索引縮圖，以及 3) 多邊形區域選取。這些展示顯示的多邊形描述是一個可以容易的就被設計成各種應用的多功能模型。這些應用包含了資料挖況，圖形識別，多媒體，與資料分群等。

除此之外，透過多邊形模型來描述諸如股市行為等的抽象資料也有助於我們觀察這些抽象的資料。也就是說，多邊形模型不光只是一個在特徵空間中的特徵向量，多邊形模型還可以告訴我們許多關於目標物的資訊。由於多邊形描述代表了資料分佈的形狀，透過觀察估算出來的多邊形描述就可以得到許多有用的資訊。

Polygon Descriptor and It's Applications

Student: Lai, Por-Shen Advisor: Prof. Fu, Hsin-Chia

Department of Computer Science

National Chiao-Tung University

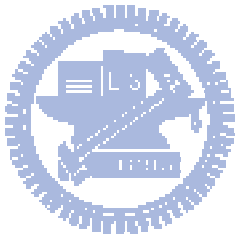
Abstract

Polygon descriptor, a polygon-based shape model, is proposed to represent the shape of a data distribution in numerical form. By sampling signals from the factors of a system, a data distribution can be drawn to show the data dependency among factors. Since data dependency implies the system behavior, abstract system behavior can therefore be represented by the proposed Polygon descriptor which models the shape of sample data distribution in numerical representation. In addition, two similarity measurement methods - 1) Deforming distance and 2) Normal vectors match, are proposed to measure the similarity between polygon descriptors. Deforming distance measures the similarity between polygon descriptors by finding a minimal-cost combination of predefined deforming operations which transform a polygon descriptor to the other. Normal vectors match finds a best mapping between two sets of normal vectors to align two polygon descriptors, and measures the similarity between Polygon descriptor according to the angle difference between normal vectors. In general, Deforming distance is much expandable but special designed for 2-D data. On the other hand, Normal vectors match is possible to be with data in any dimension but difficult to expand for specific applications.

The proposed Polygon descriptor can also contribute to similarity based clustering methods. Based on the concept of Polygon descriptor, a virtual geometry based on similarity data can be used to estimate the variance of each cluster. A demonstration, which clusters images by using the Polygon descriptor enhanced K-Medoid, shows that the proposed methods can also be useful to improve the ability of clustering methods. Besides, the coverage area of a polygon descriptor can be used as a region representation. By describing the interesting region in the polygon shape of a polygon descriptor, various multimedia application, including region selection, tracking and so on, can be carried out.

In correspondance with the three proposed extension methods, 1) measurement of shape deformation, 2) virtual geometry for similarity based clustering, and 3) polygon-based region representation, three real world application, 1) financial data mining, 2) variable-sized thumbnail of web gallery service, and 3) polygonal region selection, are also demonstrated in this dissertation. These demonstrations show that Polygon descriptor is a versatile method, which is simple to be extended in various application domains, such as data mining, pattern recognition, multimedia, data clustering and so on.

Besides, describing abstract objects, such as stock market behavior, can also help us visualize the abstract objects. That is,a polygon descriptor is not only a feature vector in feature space, but also a descriptor that tell us a lot of informations about the target object. Since a polygon descriptor represents the sample data distribution, observing a polgyon descriptor estimated from a set of sample data already reveals various information about the observed.



Acknowledgement

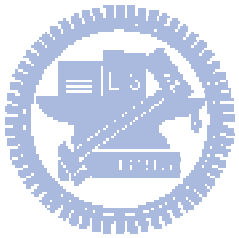
It is a pleasure to thank those who made this dissertation possible. Without supports form my advisor, 傅心家教授, I can not have a perfect environment for my research. Without suggestions from oral examiners, 劉長遠教授, 傅心家教授, 林正中教授, 陶金旭教授, 單智君教授, 詹寶珠教授, and 孫光天教授, I hardly to get so many useful information in different angles.

Thanks to all my friends. I believe that an idea comes from various incidents joined together, and the sparks of my idea comes from discussions with members of Neural Network Laboratory, including Black, Y.Y, Y.H, S.C, S.S, and so on. I have enjoyed the time when we works together.

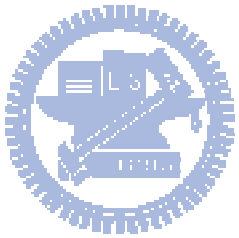
Thanks to my family. Without your support, I can not focus on my work for so many years. And your kindnesses always rescue me from my sadness.

Lastly, I offer my regards and blessings to all of those who supported me in any respect during the completion of the project.

Por-Shen Lai

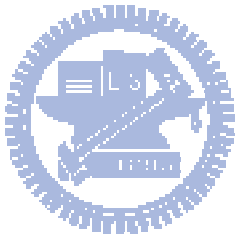


Contents



List of Tables

List of Figures



Chapter I

Introduction

1 Background

Webster defines the word "**factor**" as "*one that actively contributes to the production of a result*". A general and reasonable working definition of "**system**" is "*a set of interacting or interdependent entities, real or abstract, forming an integrated whole*".

Given an unknown system, signals are sequences of values collected by sampling the behavioral observation values in a system. Based on the working definition of system, the interactions among signals, sampled from a system, can be regarded as the behavior of a system.

As shown in Figure I.1, the relation between factors can be visualized by drawing the data distribution of sampled signals. Both of the two data distributions shown in Figure I.1 can be expressed by a line with slope equal to 0.5. That is, using a line to represent the data distribution is appropriate. However, the phenomenon that most sample data points in Figure I.1(b) converged around $(40, 40)$, can not be explained by such line models.

Since most real-world systems belong to open system, analyzing a system by inspecting all factors is almost impossible or impracticable. Thus, most analyses mainly focus on major factors only. However, ignoring minor factors

may cause the sampled signals noisy and randomness. That is, describing a data distribution by equations may not be good enough because some information contributed by the *not-considered* factors are ignored. As shown in Figure I.2, two exemplar data distributions are plotted using the stock price and earn per share(EPS) signals sampled from Taiwan Stock Market in March 1996 and March 2006.

Finding equations to describe the relation between stock price and EPS in these two example is difficult or impracticable. That is, for data distributions sampled from a real-world system, a shape-based numerical model could be more appropriate to quantitively represent the region of the distribution by its shape.

Since the shape of sample data distribution implies the relation among factors (random variables), system behavior can therefore be represented by the shape model in quantitative form.

2 Motivation and Goal

In order to do data mining on abstract system behavior, system behavior has to be represented in a numerical form first. That is, we have to translate the observed in data mining applications to a computable form. The design of such a computable form has to characterize the system behavior in a reasonable way.

System behavior results from the interaction and interdependency among acting entities, which are called **factors** in a system. For example, temperature, humidity, and so on are factors in a weather system. Stock price, earn per share (EPS), etc,. are factors in an economic system.

According to the general definition of systems, human life is related to various systems, such as weather system, economic system, ecology system,

cultural system, and so on. Thus, realizing how a system works is useful to help people make decisions, control risk and so on. Since the interaction or interdependence among factors of a system determines the **behavior** of a system, a quantitative representation of system behavior is essential to system analysis for tracking the evolution of system, comparing between systems, etc. Therefore, the motivation of this research is to design a model for capturing system behavior in numerical form and measure similarity between system behavior quantitatively.

Proposing a data model to enable the computation of data distribution is the goal of this dissertation. Although statistical quantities of a data distribution can be used for computing, there are too much details lost when measuring statistical values. On the other hand, while general data models could emulate the coverage area of data distribution well, its complexity in computation is demanding.

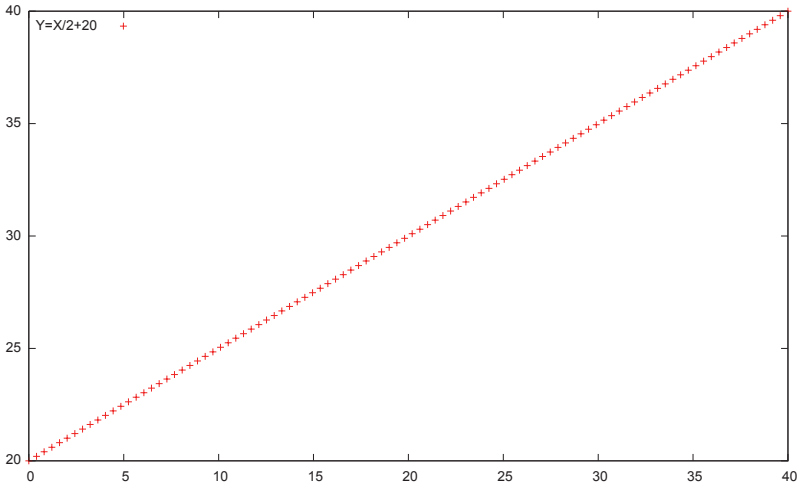
In summary, the goal of this dissertation is to design a numerical model which is capable to meet the following requirements. First of all, the model has to be suitable for representing a noisy and randomness data set. Besides, the numerical representation should allow for geometry operations. As shown as Figure I.3, the proposed model has to facilitate the work of shape comparison in account of the following four kind of representational invariance due to geometry operation - 1) translation, 2) scale, 3) aspect ratio, and 4) rotation. 1) Translation invariance means that translating the data distribution to another position can't change the shape representation; 2) Scale invariance means that resizing the data distribution can't change the shape representation; 3) Aspect ratio invariance means that resizing the data distribution without keeping its aspect ratio can't change the shape representation; 4) Rotation invariance means that rotating the data distribution

can't change the shape representation.

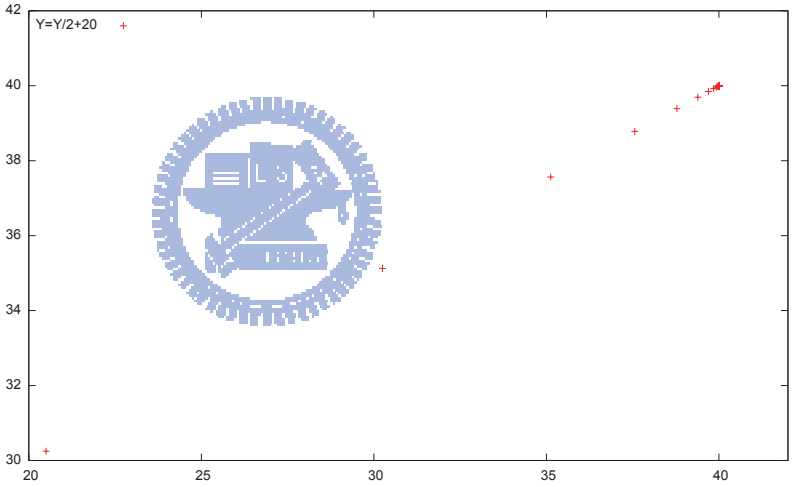
Aiming for measuring the similarity between data distributions of data sets, the shape of data distribution should be represented by an quantitative mathematical forms. Besides, to let the similarity measurement be invariant against translation, resize, and rotation operations, the location, size, and orientation of the shape of data distribution should be extractable from the proposed shape representation.

3 Dissertation Organization

This dissertation is organized as follows. At first, related works, including statistical methods and data models, are introduced in Chapter II. Then the proposed model, Polygon descriptor, is introduced, evaluated, and discussed in Chapter III. Based on Polygon descriptor, three extensions are proposed to bring the Polygon descriptor to real world applications. The first extension, measurement of shape deformation (see Chapter IV), is the major goal of this dissertation. It measures the similarity of abstract system behavior in different time periods of stock market by describing the data distribution as a polygon descriptor, and comparing two polygon descriptors by deforming distance. Polygon descriptor can also be used to improve the functionality of similarities based clustering algorithm. Section V introduces a method which estimating variance values for similarity based clusterings, such as K-medoid. Since the only available information for a similarity based clustering is the pairwise similarity between sample data, Polygon descriptor is required for estimating virtual variances of each clusters. The last extension is introduced in Section VI. Since a polygon descriptor can be used to represent a polygonal region in feature space, Polygon descriptor can also be used as the unit of tracking objects. Last, conclusion remarks are drawn in Chapter VII

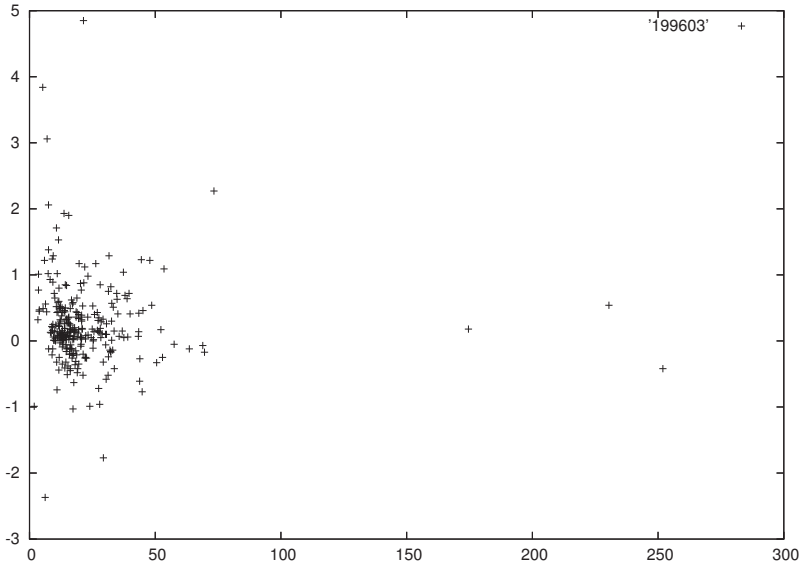


(a) $Y = X/2 + 20$

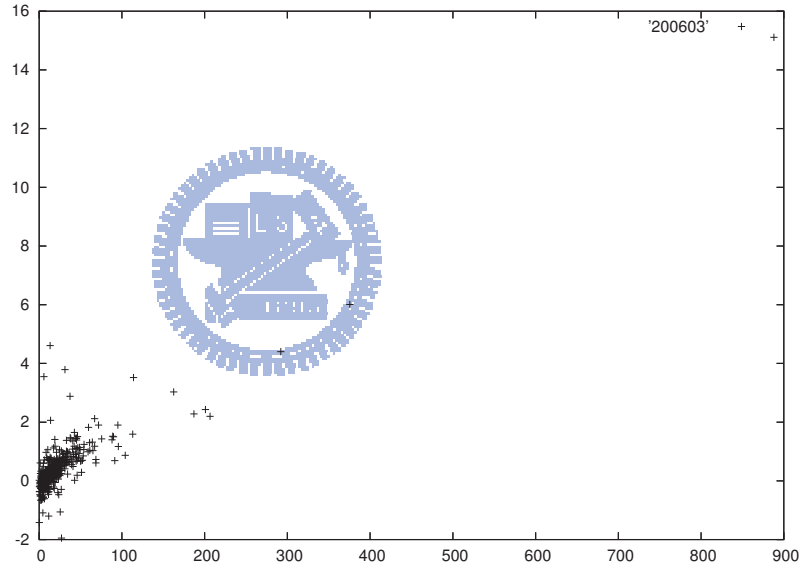


(b) $Y_i = Y_{i-1}/2 + 20$

Figure I.1: Two synthesized exemplar data distributions. (a) the data distribution between X and Y; (b) the data distribution between Y_i and Y_{i-1}.

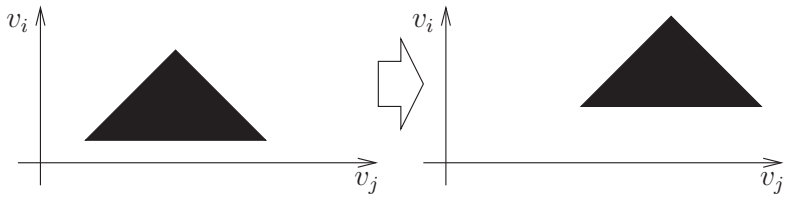


(a) March/1996

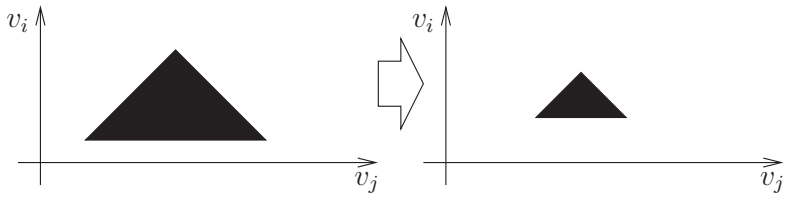


(b) March/2006

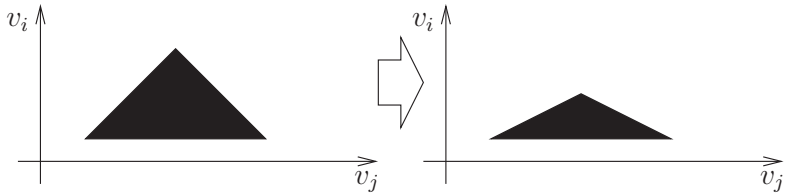
Figure I.2: Two examples of data distribution sampled in different time periods. Stock price and EPS (earn per share) in March 1996 and March 2006 are collected. As this figure shown, the market has very different behavior in March 1996 and March 2006. In 2006, the stock price is more dependent on EPS than in 1996.



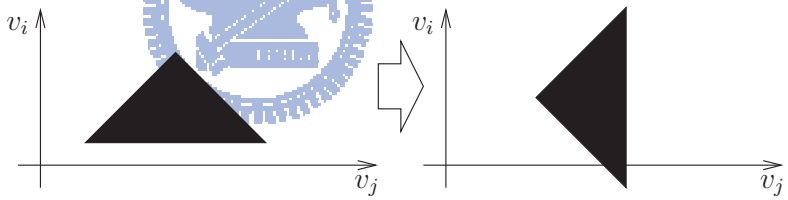
(a) Translation Invariance



(b) Scale Invariance



(c) Aspect Ratio Invariance



(b) Rotation Invariance

Figure I.3: Four type of invariance. (a) Translation Invariance - translating the data distribution to another location can't change the representation; (b) Scale Invariance - resizing the shape of data distribution can't change the representation; (c) Aspect Ratio Invariance - resizing the shape of data distribution without keeping its aspect ratio can't change the representation; (d) Rotation Invariance - rotating the shape of data distribution can't change the representation.

Chapter II

Related Works

People collect data samples to find relations among system factors. For example, we measure the falling distance and elapsed time of dropping a object to figure out how gravity system works. To show the behavior of a system, numerous methods, including statistics, data models, and so on, were proposed to represent system behavior by numbers, equations, or data models. Mostly, these methods use sample data distribution for estimation. Sample data distribution actually shows how a system works.

In this chapter, data representation methods, including statistics, and data models, are discussed. Statistical methods, including Moment, Mean, Variance, Covariance, and Correlation Coefficient, are introduced in section 1. Each statistical quantity represents a characteristic of a data distribution. Several statistical quantities could describe a data distribution. Section 2 introduces several modeling techniques, including Linear Regression, Gaussian Model, K-Means, Gaussian Mixture, and Histogram. Data models try to fit a data distribution with the coverage area by adjusting its numerical parameters. Most data models are useful to partition the (feature) space into regions for recognition applications. Last, concluding remarks are drawn in section 3.

1 Statistical Methods

By representing a factor of an unknown system by a random variable X, a signal is a sequence of experiment outcomes of random variable X. The characteristic of a factor can be described by statistic of the corresponding random variables, such as Mean, Variance, Covariance, and so on.

1.1 Moment

Given a random variable X, the moment generating function, $M(t)$, of the random variable X is defined for all real values of t by

$$M(s) = E[e^{sX}] = \int_{-\infty}^{\infty} e^{sX} f(X) dx \tag{II.1}$$

where $f(X)$ is the distribution function of X.

By differentiating Equation II.1 n times, we obtain

$$M^{(n)}(s) = E\{X^n e^{sX}\} \tag{II.2}$$

By assigning n and s to Equation II.2, moment functions are generated to measures various characteristics of a random variable. For example, when $n = 1$ and $s = 0$, $M'(0) = E[X]$, named **mean**, measures the expected value of the experiment outcomes of a random variable X.

1.2 Mean

The expected value $E[X]$ of a random variable X, named **mean**, is defined by

$$E[X] = \sum_{X:p(X)>0} Xp(X),$$

where $p(X)$ is the associated probability distribution of X. Theoretically, the average of numerous experimental outcomes of a random variable X should approximate the mean value.

1.3 Variance

Let X be a random variable and μ be the mean of random variable X. Variance of a random variable X, denoted by $var(X)$, is defined by

$$var(X) = E[(X - \mu)^2] = E[X^2] - (E[X])^2.$$

Variance indicates the distributing range of experiment outcomes of random variable.

1.4 Covariance

Covariance between two random variable X and Y, denoted by $cov(X, Y)$, is defined by

$$cov(X, Y) = E[(X - E[X])(Y - E[Y])]$$

That's, Covariance measures the variance between each pair of random variables. For example, $cov(X, Y)$ is a matrix containing four variance measurement - $E[X^2]$, $E[XY]$, $E[YX]$, and $E[Y^2]$.

Covariance is a measure of how much two variables change together. Since Covariance matrix describes the variance for each pair of random variables, a Covariance matrix can indicate the orientation of data distribution.

1.5 Correlation Coefficient

Correlation of two random variables X and Y, denoted by $\rho(X, Y)$, is defined, as long as $var(X)\, var(Y)$ is positive, by

$$\rho(X, Y) = \frac{cov(X, Y)}{\sqrt{var(X)\, var(Y)}}$$

Correlation Coefficient indicates the strength and direction of a linear relationship between two random variables.

2 Data Models

Data model uses predefined mathematical kernel, such as lines, normal distribution, etc., to approximate the coverage area of a data distribution.

2.1 Linear Regression, and Auto-regression

Linear Regression models the relation between a output signal and several input signals by a line, defined as follows,

$$Y_i = \sum_{j=1}^{N} X_{ij}\beta_j + \varepsilon_i,$$

where Y_i is the i-th element of output signal, X_{ij} is the i-th element of the j-th input signal, and the coefficients β_j are estimated to minimize ε_i. Since the coefficients β_j show how the input signals X compose the output signal Y, the coefficients β_j can be used to describe the relation between input signals and output signal.

On the other hand, auto-regression models the relation between current element and the latest N elements in a signal by the following Equation,

$$Y_t = \sum_{i=1}^{N} \phi_i Y_{t-i} + \varepsilon_t$$

where ϕ_i are the auto-regression coefficients, Y_t is the signal under investigation, and N is the order(length) of the filter which is generally very much less than the length of the series.

2.2 Gaussian Model

Unlike linear regression method, Gaussian model[14] uses Gaussian/Normal distribution to describe the data distribution of input signals. Gaussian/Normal distribution $f(x)$ with parameters μ and σ^2 is given by

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

According to Bayes' rule, probability density functions of Gaussian model can be used to measure the probability of a particular class C_j given as input pattern x. That is, given a input pattern x, the probability $P(C_j|x)$ of a particular class C_j, corresponding to the Gaussian model, can be estimated as follows,

$$P(C_j|x) = P(x|C_j)\frac{P(C_j)}{P(x)}$$

where $P(C_j|x)$ is the probability that an input vector x belongs to class C_j; $P(x|C_j)$ is the probability density function of an input vector x if the class was known to be j, $P(C_j)$ is the prior probability for class j, and $P(x)$ is the overall probability density function of x.

The unimodal Gaussian method generates an estimate of $P(x|C_j)$ as follows,

$$P(x|C_j) = \frac{1}{(2\pi)^{n/2}|V_j|^{1/2}} e^{[-1/2(x-M_j)^T V_j^{-1}(x-M_j)]}$$

Free parameters include the mean M_j of input vectors of each class and the covariance matrix V_j of each class j.

2.3 K Means

K-Means model[11] uses K mean points to represent a data distribution. K mean points are estimated by minimizing

$$\sum_{i=1}^{N} \min_{\bar{x}_k \in \bar{X}} \|x_i - \bar{x}_k\|^2$$

where $\bar{X} = \{\bar{x}_1, ..., \bar{x}_K\}$ is a set of mean vectors.

The training algorithm[15] is stated as follows:

1. Initial K mean vectors $\bar{x}_k$, for $k = 1, ..., K$, by randomly selecting from training data points.

2. Compute cluster assignments $C(x_i)$ of data points $x_i \in X$, where X is the training data set, by

$$C(x_i) = \underset{\bar{x}_k \in \bar{X}}{\operatorname{argmin}} \|x_i - \bar{x}_k\|^2.$$

where $\bar{X}$ is the set of mean vectors.

3. Measure the means vectors $\bar{x}_k \in \bar{X}$ by using the cluster assignments $C(x_i)$ and the following equation,

$$\bar{x}_k := \frac{1}{N_k} \sum_{C(x_i)=\bar{x}_k} x_i,$$

where N_k is the number data points with cluster assignment $C(x_i) = \bar{x}_k$.

4. Repeat steps 2 and 3 until the mean vectors converged.

2.4 Gaussian Mixture Model

Gaussian Mixture model[16] combines multiple Gaussian models to describe the coverage area of a data distribution in arbitrary shape. A Gaussian Mixture Model is defined as follows,

$$P(x|C) = \sum_{k=1}^{N_c} w_k G_k$$

where w_k is the weight for the k-th Gaussian G_k, $\sum_{k=1}^{N_c} w_k = 1$, and a Gaussian component G_k is defined as:

$$G_k = \frac{1}{(2\pi)^{n/2}|V_k|^{1/2}} e^{[-1/2(X-M_k)^T V_k^{-1}(X-M_k)]}$$

where M_k and V_k are the mean and covariance matrix of the k-th Gaussian component respectively.

Generally, parameters, including weight, mean, and covariance matrix, of a Gaussian Mixture Model are estimated using an iterative procedure called EM algorithm[5]. EM algorithm maximize the likelihood L or log-likelihood $ln(L)$ between a given training set $X = \{x_1, ..., x_{N_{train}}\}$ and the probability density function of a Gaussian Mixture Model. The likelihood function L for a class C is defined as,

$$L = \prod_{i=0}^{N_{train}} P(x_i|C).$$

The log-likelihood is defined as,

$$ln(L) = \sum_{i=0}^{N_{train}} ln(P(x_i|C)).$$

where N_{train} is the number of sample data points, $P(x_i|C)$ is the probability of a data point x_i in C.

The training algorithm of Gaussian Mixture Model with K Gaussian is stated as follows:

1. Initialize the K Gaussian means $\mu_i, i = 1, ..., K$ by using K-Means algorithm.

2. Initialize the K Gaussian covariance matrices V_i to the distance to the nearest cluster and the weights π_i to $1/K$.

3. Let the probability of a data point x in the i-th Gaussian component G_i be

$$p(x|G_i) = \frac{1}{(2\pi)^{d/2}|V_i|^{1/2}} e^{[-1/2(x-\mu_i)^T V_i^{-1}(x-\mu_i)]},$$

where μ_i and V_i are the mean and covariance of a Gaussian component G_i. d is the dimension of x.

4. Compute the probability $p(G_i|x)$ for each Gaussian component G_i, for

$i = 1...K$ in condition of a given data point x.

$$p(G_i|x) = \frac{\pi_i p(x|G_i)}{p(x)} = \frac{\pi_i p(x|G_i)}{\sum_{i=1}^{G} \pi_j p(x|G_j)}$$

5. Update the weights π_i, means μ_i and covariance matrices V_i for each Gaussian component G_i:

$$\pi_i := \frac{1}{N_c} \sum_{x \in X} p(G_i|x),$$

$$\mu_i := \frac{1}{N_c \pi_i} \sum_{x \in X} p(G_i|x) x,$$

$$V_i := \frac{1}{N_c \pi_i} \sum_{x \in X} p(G_i|x) \left((x - \mu_i)(x - \mu_i)^T \right),$$

where X is the training data set and N_c is the number of training data points in X.

6. Repeat step 3,4,5 until π converges.

2.5 Histogram

As the example shown in Figure II.1, a feature space is partitioned into several bins. The number of data points in each bins are used to represent the data distribution. By referencing the maxima and minima in each dimension, a feature space is partitioned into several bins. By counting the number of data points in each bins, a histogram is constructed. Since the location of bins are based on the the maximal and minimal value in each dimension, representing a data distribution by a histogram is capable to be invariant to translation, scale, and aspect.

2.6 Polar Histogram

As the example shown in Figure II.2, Polar histogram[9] can be invariant to rotation by segmenting a feature space by radial cuts and concentric circles.

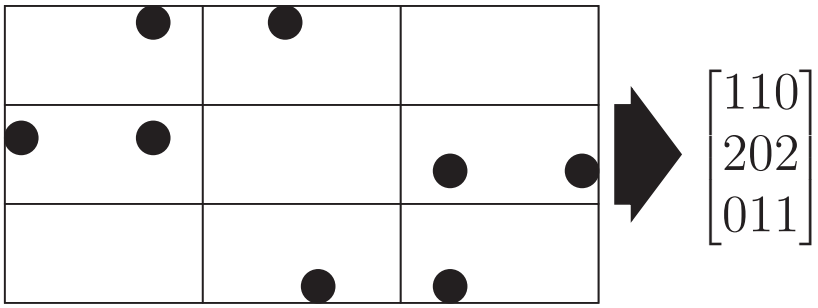
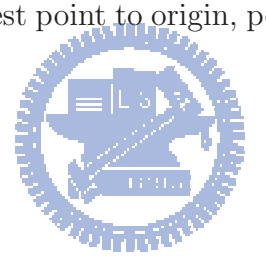


Figure II.1: According to the maximal and minimal value in each dimension, a feature space is partitioned into several bins. By counting the number of data points in each bins, a histogram is constructed to represent the original data distribution.

That is, the data bins of Polar histogram is decided according to the radius δ and the angle α. Since Mean is subtracted from data points, polar histograms are invariant to *translation*. Since the radius δ is normalized by the distance from farthest elements to Origin, polar histogram is also invariant to *scale*. Since α is the angel between the line from the data point to origin and the line from the farthest point to origin, polar histogram is invariant to *rotation*, too.



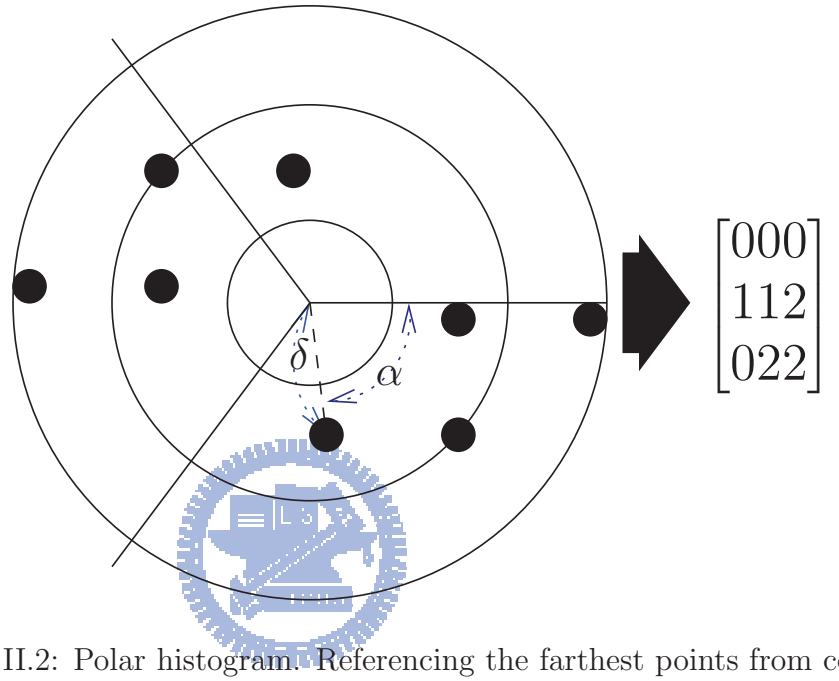


Figure II.2: Polar histogram. Referencing the farthest points from center, a disc-shaped partition is created to computing the polar histogram.

3 Concluding Remarks

In general, representing a data distribution by several statistical values, such as Mean, Variance, and so on, dropped too much details. Besides, since most statistical values represents the global characteristic of a sample distribution, characteristics of local regions are usually unattended.

On the other hand, data models, such as Gaussian Mixture, K-Means, and so on, try to create a best fit coverage area for sample data distribution by mixing components. That is, by using a data model which is estimated from a set of sample data points $\mathcal{A}$, we can examine whether a test data is contained in the region expanded by sample data $\mathcal{A}$. However, since a data model is usually a mixture of several kernel models, extracting characteristics of data distribution from a data model is often difficult.

In order to describe the shape of a data distribution, and reveal characteristics of a data distribution, a statistical shape model, which uses a simple single structure to represent the shape of a sample data distribution, is required. Without fitting the region of a data distribution, a model may lost too much details of sample distribution. Without using a simple single structure, a model may be difficult to analyze or to make extensions. Thus, Polygon descriptor is proposed in Chapter III to model a data distribution in terms of a single generalized polygonal region.

Chapter III

Polygon Descriptor

In general, linear or non-linear mathematical model is very difficult to represent the randomness of data distribution, because they describe a system by analytical functions. Thus, modeling data by complex mixture models are proposed to preserve more inherent characteristics of a data distribution. However, mixture models often lack of operational flexibilities, such as translation, scale, and rotation invariant operations, which are useful in computation of shape difference. In the past, shape analysis[20][29] methods with transform, scale, and rotation invariance were proposed for similarity measurement among images. However, most of these methods require shape features, such as edges or boundaries to represent an object or a data distribution.

In this chapter, a polygon-based shape model, named Polygon descriptor, is proposed to represent a random and noisy data distribution in a translation, scale, and rotation invariant manner. In the following, we first present the Polygon descriptor in a formal mathematical form.

1 Model Definition

Polygon descriptor is proposed to formulate a data distribution. Given a set of data points, a polygon descriptor can be used as feature objects to characterize the dependencies among data variables. Since Polygon descriptor represents the geometry of a data distribution in numerical form, the feature represented by a polygon descriptor is useful for translation, scaling, and rotation invariant comparisons.

In order to extend the Polygon descriptor concepts to any dimensionality, a **generalized Polygon** is first defined as follows:

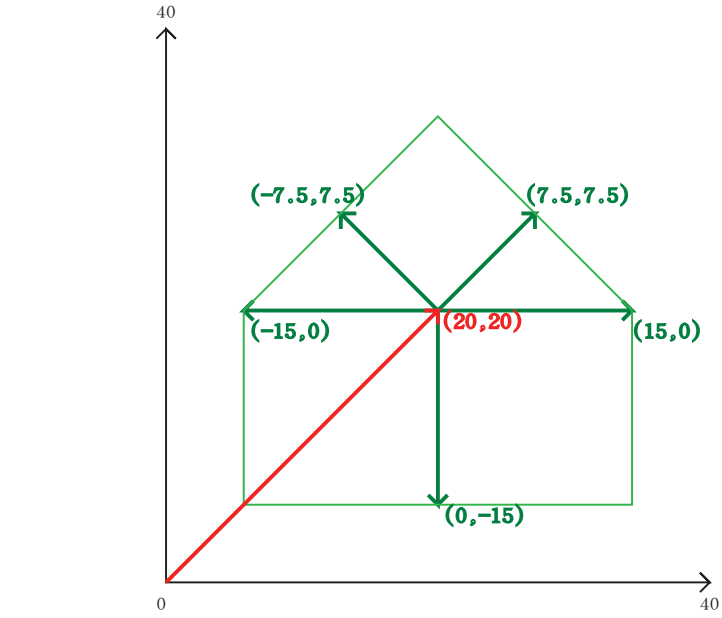
1. A generalized Polygon is an union of several **Convex Unit**s.
2. A Convex Unit is a hyperspace surrounded by several hyperplane.
3. A Polygon descriptor is the mathematical formulation of a convex unit.

A **Polygon descriptor** contains a reference center and N normal vectors. A normal vector represents: 1) the normal direction of a hyperplane which encloses the convex unit, and 2) the distance from the reference center to each hyperplane. Figure III.1(a) shows an exemplar Polygon descriptor for the representation of a 2D data distribution in a convex unit. The reference center is located at $(20, 20)$ and five normal vectors are $\begin{pmatrix} 15 \\ 0 \end{pmatrix}$, $\begin{pmatrix} 7.5 \\ 7.5 \end{pmatrix}$, $\begin{pmatrix} -7.5 \\ 7.5 \end{pmatrix}$, $\begin{pmatrix} -15 \\ 0 \end{pmatrix}$, and $\begin{pmatrix} 0 \\ -15 \end{pmatrix}$. By binding a 1-D probability function to each normal vectors, a polygonal probability model is created as shown in Figure III.1(b). Suppose the 1-D probability function is a Gaussian function, then the polygonal probability model (distribution) can be centered at $(20, 20)$ and the variance corresponds to each normal vector can be the length of each normal vectors respectively.

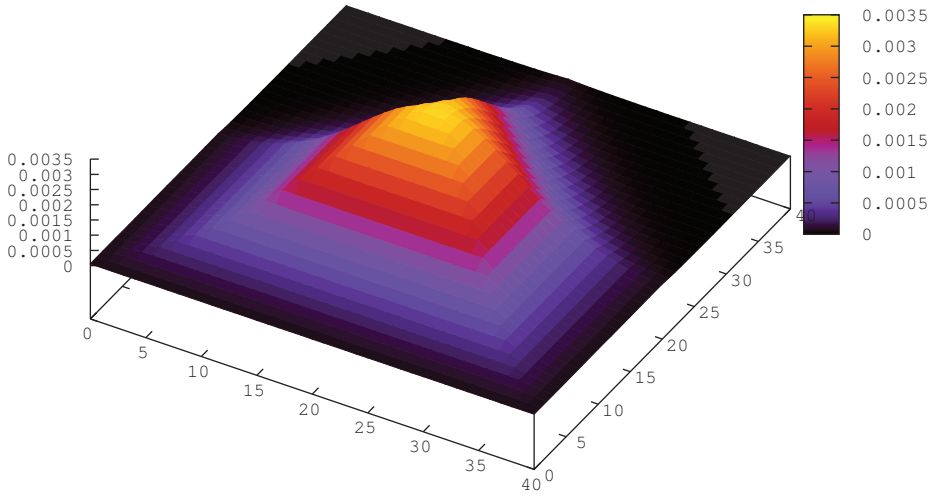
By combining several polygon descriptors, a generalized Polygon can be represented by the proposed mathematical form. By using a generalized Polygon to model a set of data points, the data points may often need to be clustered into several groups (convex units) according to the requirements of applications. Since a single convex unit is enough to describe the data dependencies among data variables, which is the motivation of this dissertation, this section will focus on the case of single convex unit only.

Symbols related to Polygon descriptor are defined as follows.

$\vec{C}$	**Reference Center**. The Medoid among a set of data points are selected to be the reference center.
S_j	**Pseudo boundary**. A set of imagine hyperplane which wraps the distributed area of a data set. The space wrapped by pseudo boundaries should reflect the shape of data distribution.
$\vec{a_j}$	**Normal Axes**. A set of data vectors starting from the reference center to the pseudo boundaries of data distribution. Basically, a normal vector a_j is orthogonal to a pseudo boundary S_j.
$\vec{G_j}$	**Side Cluster**. As shown in Figure III.2, a side cluster contains data points located in a pyramidal space defined by a tip point at C and a bottom side S_j.
α	**Normal-to-Boundary ratio**. As shown in Figure III.3, Normal-to-Boundary ratio is defined by $\frac{\text{the length of a normal vector } a_j}{\text{the length of a pseudo boundary } s_j}$.



(a)



(b)

Figure III.1: (a) An exemplar of Polygon descriptor, where its center is at $(20, 20)$ and normal vectors (drawn as solid arrow) to each surrounding hyperplane are $(15, 0)^T$,$(7.5, 7.5)^T$,$(-7.5, 7.5)^T$,$(-15, 0)^T$, and $(0, -15)^T$. (b) The pentagonal probability distribution for the polygon descriptor of (a).

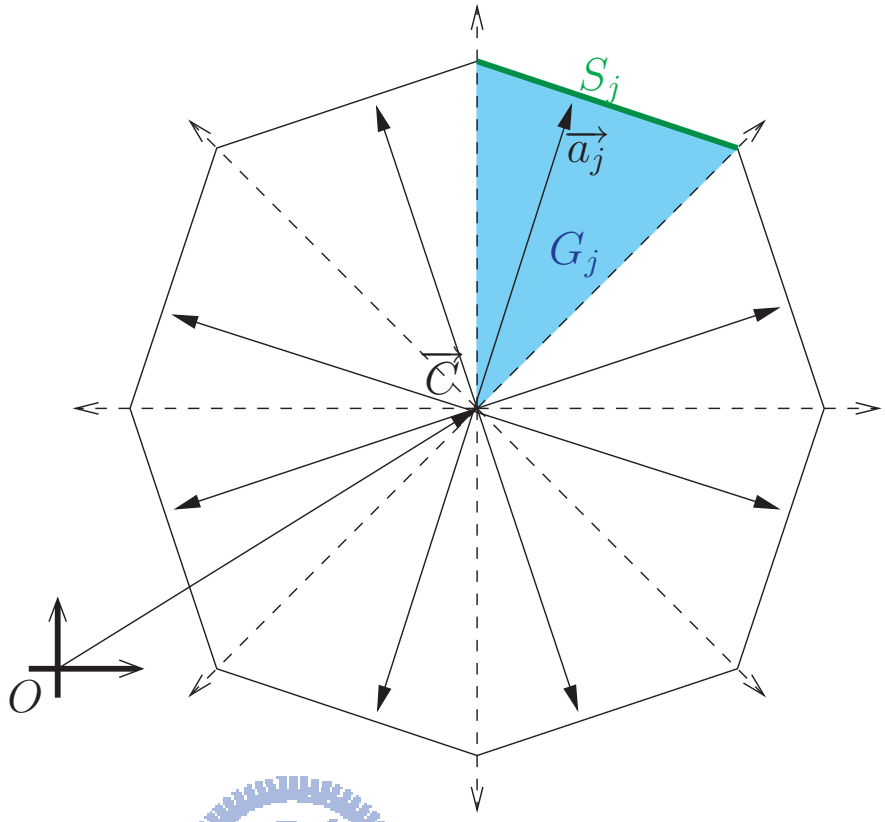


Figure III.2: An exemplar Polygon descriptor is given to show the meaning of symbols related to Polygon descriptor.

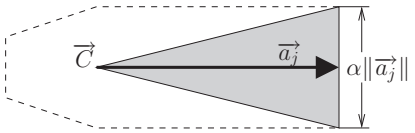


Figure III.3: An exemplar side cluster is given to show the meaning of Normal-to-Boundary ratio. A smaller Normal-to-Boundary ratio represents a narrower side cluster.

2 Model Estimation

To describe the data distribution for a set of data points by a polygon descriptor, a learning (fitting) algorithm is proposed to iteratively approximate the shape of data distribution by adjusting the model parameters.

2.1 An Idea of Polygon Description Estimation

Figure III.4 depicts a data distribution D_a resulting from a uniform distribution in a diamond region D_b and a uniform distribution in a square region D_c. The dashed lines in Figure III.4 show the desired boundaries of data distributions D_a, D_b, and D_c.

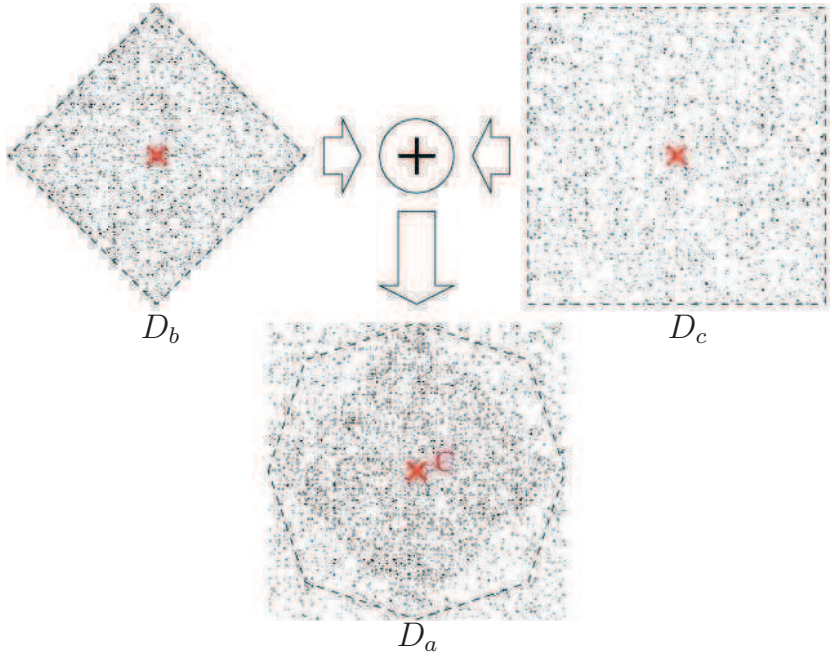


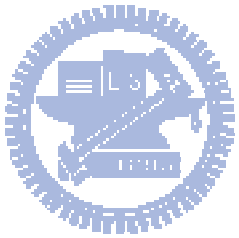
Figure III.4: A data distribution may be composed by several components in different shape. In this example, a data distribution D_a is consisting of a uniform distribution in a diamond region D_b and a uniform distribution D_c. The boundaries for D_c is supposed to be the average boundaries of its two components.

By plotting such composite component in 3-D layout, the density function will like Figure III.5. The z-axis implies the density in everywhere. Figure III.5 (A) draws the density function of data distribution D_a given in Figure III.4. A radial cut θ from the center point c is shown in Figure III.5 (A.1) and (A.2). As shown as the example, the data distribution can be decomposed into several horizontal density layers (red line). s_i is the distance from center to distribution boundary for each density layer. a_i is the average distance from center to data points in each density layer. The example also shows that $s \approx ka$ where k is a constant when the data dimension is fixed. Another example for a data distribution consisting of infinite number of density layers is given in (B), (B.1), and (B.2). Since the ratio between s and a is fixed, a down-sized shape for a data distribution can be estimated by connecting the average distance from center to data points in every radial cuts. That is, the shape of data distribution can be approximated by 1) partitioning the feature space into segments by using hyperplane passing through the center point, 2) computing the mean points for each segments and 3) connecting mean points of neighboring segments to represent the shape of data distribution.

Therefore, the average boundaries of each uniform components are proposed to represent the shape of a data distribution, which may be composed by various uniform distributed regions.

Figure III.6 shows another example. The density in dark gray region is higher than in light dark region. Besides, the high density region is not connected. Therefore, with a reference center in one high density peak, the other one will cause a raising part in the resulted contour. Such results are quite nature, because there is supposed to have a spread over that direction where plenty of data samples, far apart from the reference, are distributed along the direction.

On the other way, another observation is required to develop the learning (fitting) algorithm for Polygon descriptor. Figure III.7 uses an example to explain the Polygon descriptor learning method introduced in this section. By partitioning the feature space by two hyperplane $x = 0$ and $y = 0$, the estimated polygon contour is the solid square. By partitioning the feature space into more segments with two extra hyperplane $y = x$ and $y = -x$, the estimated polygon shape is drawn in dashed lines. Apparently, the estimated polygon approximates the shape of data distribution better when more hyperplane are used to segment the feature space. However, the robustness of mean estimation for each segment is related to the size of segments. That is, to be robust to noisy data, the size of segment has to be large. Therefore, much delicate learning algorithm is required to estimate a polygon descriptor to approximate the shape of data distribution.



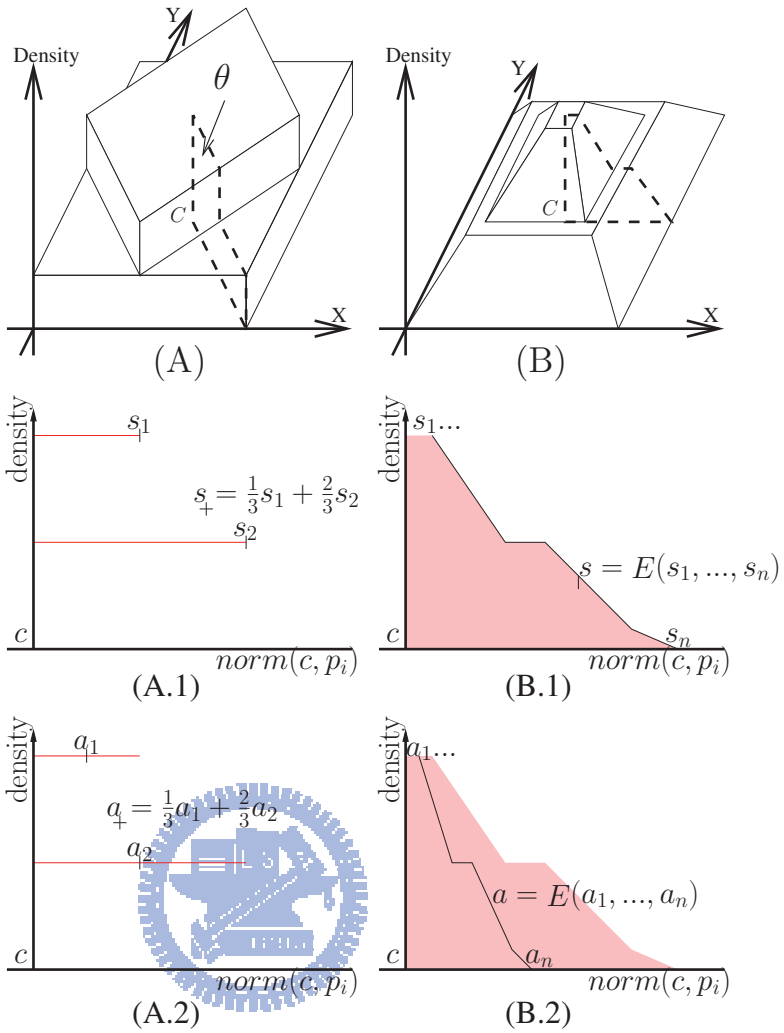


Figure III.5: The 3-D plot of two composite uniform regions. The density function of the data distribution D_a given in Figure III.4 is drawn in bird's view (A). A radial cut θ from the center point c is shown in (A.1) and (A.2). The density function can be considered as a composition of multiple density layer (red line). s_i is the distance from center to distribution boundary for each density layer. a_i is the average distance from center to data points in each density layer. As the example shown, $s = ka$ where k is a constant when the data dimension is fixed. (B), (B.1), and (B.2) are another set of examples with infinite number of density layers.

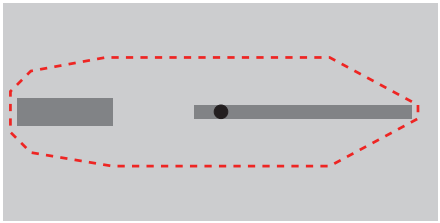


Figure III.6: Density in dark gray region is higher than in light gray region. The high density part results the contour sharp in left/right side. The disconnected high density region in the left part push the contour more left, and gap between two high density region results the contour in the left side much more similar to the contour of light gray region.

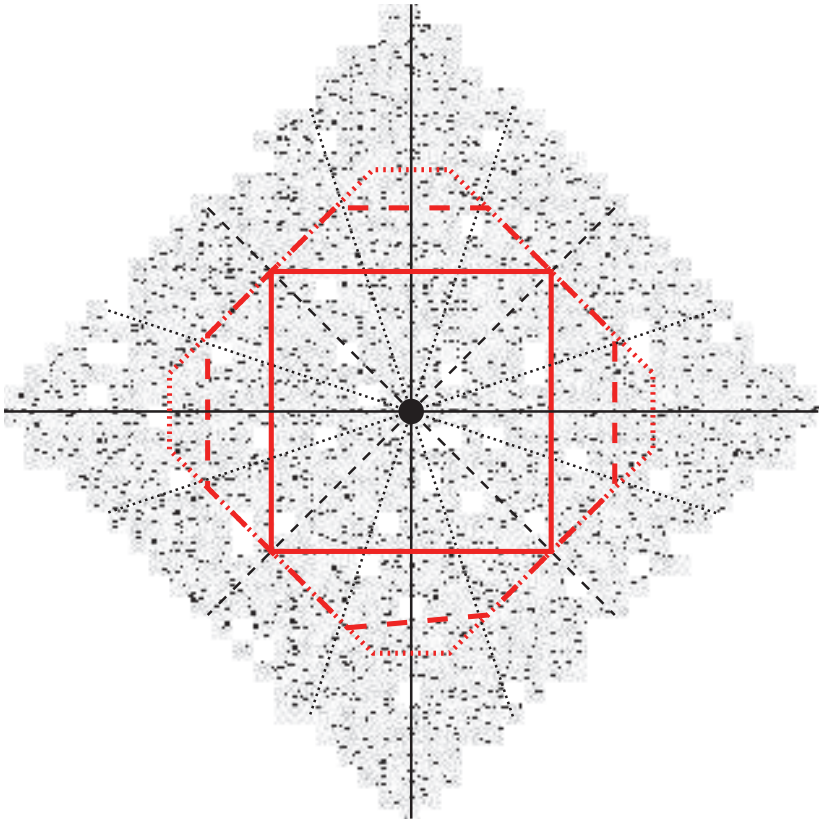


Figure III.7: An observation about the partial statistics of data clustering in pyramidal region. An example is given to introduce a simple method for the learning of Polygon descriptor. By using two hyperplane to segment the feature space into four segments, the solid square is estimated. When increasing the number of segments, the estimated polygon approximates the shape of data distribution better (dashed and dotted contours).

2.2 Learning Algorithm

According to the statistical characteristics of a data distribution, a **Polygon descriptor** can be estimated from a data distribution by an iterative process. Figure III.8 shows the flow chart of a polygon descriptor learning process. First, a reference center is estimated, and N normal vectors ,$A = \{a_1, ..., a_j, ..., a_N\}$, is initialized in a random manner. According to these normal vectors, data points are clustered into groups (side clusters) in associated with each normal vectors in A. For each side cluster, a normal vector is estimated by using data points in each side cluster. Two processes - 1) cluster data points according to normal vectors, and 2) estimate normal vectors according to the result of data clustering, are repeatedly applied until the orientation of normal vectors converges. Last, the length of normal vectors are estimated. As stated in Section 2.3, the number of normal vectors N can be determined by gradually increasing the number of normal vectors until the number of distinguishable normal vectors converges. In the followings, the learning algorithm for Polygon descriptor is presented in four parts: (1) Reference center estimation, (2) Data point clustering, (3) Normal vector orientation estimation, and (4) Normal vector length estimation.

(1) *Reference Center Estimation*

The reference center c of a polygon descriptor is defined as the Medoid of a set of data points P. According to the definition of Medoid, the reference center of a polygon descriptor can be estimated as follows,

$$c = \underset{p_i \in P}{\textbf{argmin}} \left(\sum_{p_j \in P} norm(p_i, p_j) \right), \tag{III.1}$$

where p_i and p_j are two data points in P and $norm(p_i, p_j)$ is the norm from p_i to p_j.

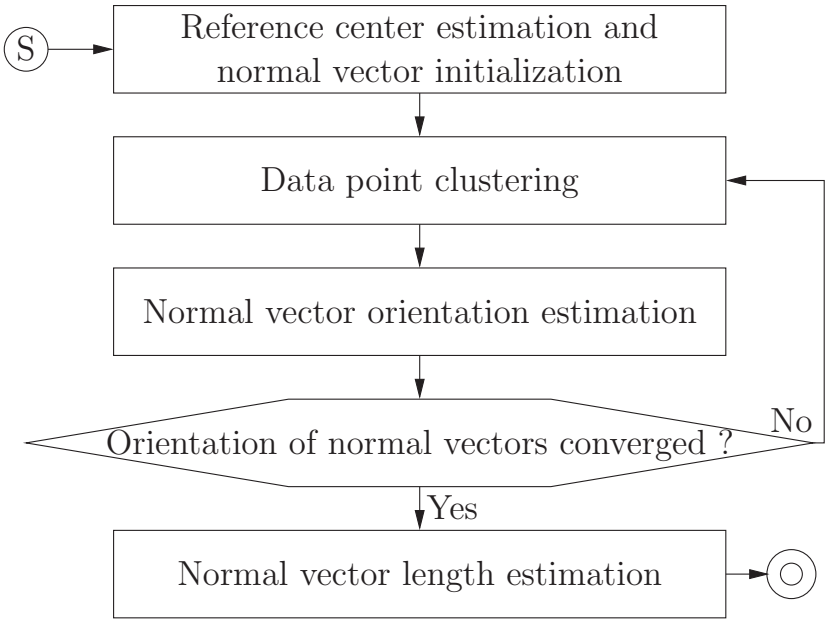


Figure III.8: The flow chart of Polygon descriptor learning process. Given a set of data points, the learning process finds a reference center and normal vectors which represents the polygon descriptor that can be best fitted to the data distribution of training data set.

(2) *Data Points Clustering*

Data point clustering process partitions data points into side clusters according to N normal vectors. That is, the data point clustering process associates each data point p_i to a normal vector $\vec{a_w}$, in the normal vector set $A = \{\vec{a_1}, ..., \vec{a_j}, ..., \vec{a_N}\}$. The associated normal vector for a data point p_i can be measured by

$$\vec{a_w} = \underset{\vec{a_j} \in A}{\mathbf{argmax}} \frac{\vec{a_j} \cdot \vec{cp_i}}{\|\vec{a_j}\|^2}, \tag{III.2}$$

where $\vec{cp_i}$ is the vector from reference center to a data point p_i.

In Eq. III.2, $\vec{a_w}$ is the normal vector that causes the largest projection length $\frac{\vec{a_j} \cdot \vec{cp_i}}{\|\vec{a_j}\|^2}$. As shown in Figure III.9, projection ratio is the ratio between the projection length $\frac{\vec{a_j} \cdot \vec{cp_i}}{\|\vec{a_j}\|}$ and the length of the normal vector $\|a_j\|$. Since a

data point p_i having identical projection ratios on two distinguishable normal vectors $\vec{a_1}$ and $\vec{a_2}$ satisfy the following Equation,

$$\frac{\vec{a_1} \cdot \vec{cp_i}}{\|\vec{a_1}\|^2} - \frac{\vec{a_2} \cdot \vec{cp_i}}{\|\vec{a_2}\|^2} = 0$$
$$\Rightarrow \quad \left(\frac{\vec{a_1}}{\|\vec{a_1}\|^2} - \frac{\vec{a_2}}{\|\vec{a_2}\|^2} \right) \cdot \vec{cp_i} = 0.$$

Data points having the same projection ratio on $\vec{a_1}$ and $\vec{a_2}$ are located at a hyperplane passing through the reference center and orthogonal to the vector $\left(\frac{\vec{a_1}}{\|\vec{a_1}\|^2} - \frac{\vec{a_2}}{\|\vec{a_2}\|^2}\right)$. Therefore, the clustering process divides a convex unit into N pyramid regions, where N is the number of normal vectors.

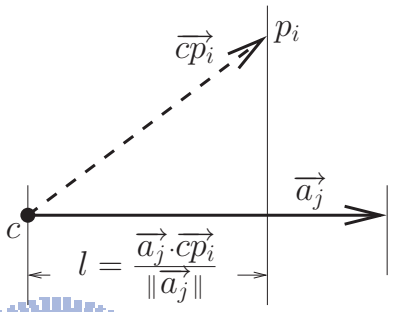


Figure III.9: The projection ratio for a data point p_i on a normal vector $\vec{a_j}$ can be calculated by dividing the projection length l by the length of a normal vector a_j.

When the normal vector is point to outside of the pyramid data distributing area, the projection ratio may become negative. However, the relation, between projection ratios, still holds.

(3) *Normal Vector Orientation Estimation*

Data points in a side cluster can be used to estimate the orientation of a normal vector.

As shown in Figure III.11, data points associated to a normal vector $\vec{a}$

are displayed in a gray-scale region. A hyperplane H, which passes through the reference center c, partitions a side cluster into two segments, I and II. Let μ_1 and μ_2 be mean points of segments I and II respectively. According to Similar Triangle properties (see Figure III.10), the line that connects μ_1 and μ_2 is supposed to be orthogonal to the normal vector $\vec{a}$. Therefore, for 2-D data distribution, the orientation of a normal vector can be estimated as follows:

1. Partition the cluster into 2 segments by a hyperplane passing through reference center;
2. Computing mean points, μ_1 and μ_2 for each segments;
3. Find a vector which is orthogonal to $\overrightarrow{\mu_1\mu_2}$.

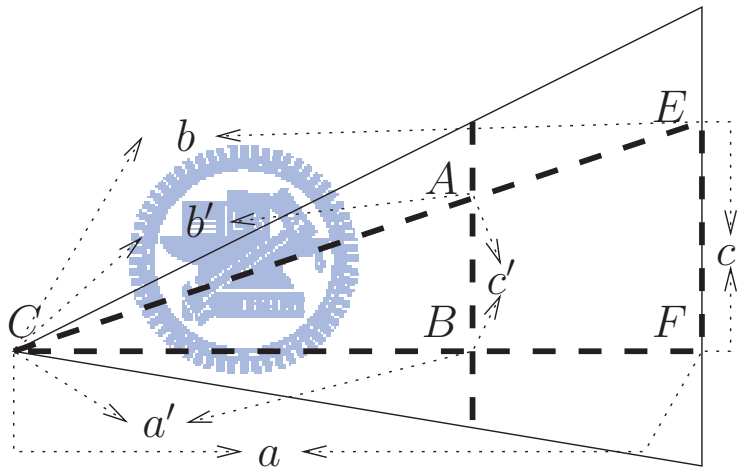


Figure III.10: An example to depict the similar triangle property. Since $\frac{a}{a'}=\frac{b}{b'}=\frac{c}{c'}$, $\triangle CAB$ and $\triangle CEF$ are similar and $\overline{AB}$ and $\overline{EF}$ are parallel.

A generalized normal vector estimation for higher dimension data distribution are introduced as followings.

When segmenting feature space using hyperplane passing through the reference center, a value R_a^b which represents the average distance between

the reference center and data points can be measured for each segment by the following Equation.

$$R_a^b = \frac{\int_{\theta=a}^{b}\int_{l=0}^{\infty} l\cos(\theta)\, pdf(l,\theta)\mathrm{d}l\mathrm{d}\theta}{\int_{\theta=a}^{b}\int_{l=0}^{\infty} pdf(l,\theta)\mathrm{d}l\mathrm{d}\theta}$$

where $\vec{p_i}$ is the vector from reference center to point p_i. l and θ are the length of $\vec{p_i}$ and the angle between $\vec{p_i}$ and X-axis, respectively. $pdf(l,\theta)$ is the probability of a point located at (l,θ).

Based on the similar triangle property, for two segments, $a < \theta < b$ and $c < \theta < d$, sharing the same bottom side, R_a^b is equal to R_c^d. That is,

$$\forall a < b \text{ and } c < d,\ R_a^b = R_c^d.$$

Since, the average distances R_a^b for every segments are the same, weighting data points by any function of angle doesn't change the value of R_a^b. That is, the angle between a normal vector and X-axis can be estimated by minimizing the following Equation,

$$Q(\theta) = \left[\frac{\sum_i f(\omega_i) l_i \cos(\omega_i - \theta)}{\sum_i f(\omega_i)} - \frac{\sum_i g(\omega_i) l_i \cos(\omega_i - \theta)}{\sum_i g(\omega_i)}\right]^2,$$

where $f(w_i)$ and $g(w_i)$ are two arbitrary functions of angle.

By expanding this objective function $Q(\theta)$, the θ minimizing $Q(\theta)$ can be computed as follows,

$$\begin{aligned} Q(\theta) &= \left[\frac{\sum_i f(\omega_i) l_i \cos(\omega_i - \theta)}{\sum_i f(\omega_i)} - \frac{\sum_i g(\omega_i) l_i \cos(\omega_i - \theta)}{\sum_i g(\omega_i)}\right]^2 \\ &= \left[\frac{\sum_i f(\omega_i) l_i \cos\omega_i \cos\theta + \sum_i f(\omega_i) l_i \sin\omega_i \sin\theta}{\sum_i f(\omega_i)} - \right. \\ &\quad \left. \frac{\sum_i g(\omega_i) l_i \cos\omega_i \cos\theta + \sum_i g(\omega_i) l_i \sin\omega_i \sin\theta}{\sum_i g(\omega_i)}\right]^2. \end{aligned}$$

Let

$$\begin{aligned} a &= \frac{\sum_i f(\omega_i) l_i \cos\omega_i}{\sum_i f(\omega_i)} - \frac{\sum_i g(\omega_i) l_i \cos\omega_i}{\sum_i g(\omega_i)} \\ b &= \frac{\sum_i g(\omega_i) l_i \sin\omega_i}{\sum_i g(\omega_i)} - \frac{\sum_i f(\omega_i) l_i \sin\omega_i}{\sum_i f(\omega_i)} \end{aligned}$$

$$\begin{aligned} Q(\theta) &= [a\cos\theta + b\sin\theta]^2 \\ &= \left[\sqrt{a^2+b^2}\cos\left(\theta - \tan^{-1}\frac{b}{a}\right)\right]^2 \end{aligned}$$

When $\theta = \pm\frac{\pi}{2} + \tan^{-1}\frac{b}{a}$, $Q(\theta) = 0$.

To generalize the solution for data points in any dimension, a vector $(1, n_2, \ldots, n_k)$ which is orthogonal to the side can be computed by solving the following equations array.

$$\begin{array}{rrcrcl} \sigma(1,1) & +\sigma(1,2)n_2 & +\ldots & +\sigma(1,k)n_k & = & 0 \\ \sigma(2,1) & +\sigma(2,2)n_2 & +\ldots & +\sigma(2,k)n_k & = & 0 \\ & & \vdots & & & \\ \sigma(k-1,1) & +\sigma(k-1,2)n_2 & +\ldots & +\sigma(k-1,k)n_k & = & 0 \end{array}$$

where $\sigma(i,j) = \frac{\sum f_i p_j}{\sum f_i} - \frac{\sum g_i p_j}{\sum g_i}$. That is, the orientation of normal vectors can be calculate by using the following Equation,

$$\begin{bmatrix} n_2 \\ \vdots \\ n_k \end{bmatrix} = \begin{bmatrix} \sigma(1,2) & \ldots & \sigma(1,k) \\ \vdots & \ddots & \vdots \\ \sigma(k-1,2) & \ldots & \sigma(k-1,k) \end{bmatrix}^{-1} \begin{bmatrix} \sigma(1,1) \\ \vdots \\ \sigma(k-1,1) \end{bmatrix}.$$

In this dissertation, Sigmoid functions are applied to measure the orientation of normal vectors for Polygon descriptor. For example, $\frac{1}{1+e^{\theta}}$ and $\frac{1}{1+e^{-\theta}}$ are applied for 2-D data points; $\frac{1}{1+e^{\theta_1}}$, $\frac{1}{1+e^{-\theta_1}}$, and $\frac{1}{1+e^{\theta_2}}$ are applied for 3-D data points. In order to use coordinates of data points directly instead of

angles, a Sigmoid function of cosine values as shown as follows can be used instead.

$$1 - \frac{1}{1 + exp(\frac{1-\vec{p_i}\cdot\vec{a_j}}{\vec{a_j}\cdot\vec{a_j}})}$$

where $\vec{a_j}$ is the normal vector of a side cluster S_j and $\vec{p_i}$ is a vector from reference center c to a data point p_i.

(4) *Mean Length of a Normal Vector Estimation*

To draw a boundary or outlines of a data distribution is often not feasible or in fact is not very meaningful. Thus to estimate the length of a normal vector is also meaningless. Hence, we propose to estimate the mean length of a normal vector, so that the approximated shape of a data distribution can be visualized. The mean length of a normal vector are adjusted as follows,

$$\|\vec{a_j}\| := \frac{\vec{a_j}\cdot\vec{\mu_j}}{\sqrt{\vec{a_j}\cdot\vec{a_j}}},$$

where $\vec{\mu_j}$ is the mean vector estimated using all data points associated to $\vec{a_j}$. That is, the mean length of normal vector is adjusted to be equal to the projection length of the mean vector along the normal vector.

As shown in Figure III.11, the pentagon drawn in solid line illustrate the Polygon descriptor which holds the approximated shape of a given data distribution. In fact, the data points can not be enclosed by any distinct distinct boundary hyperplane.

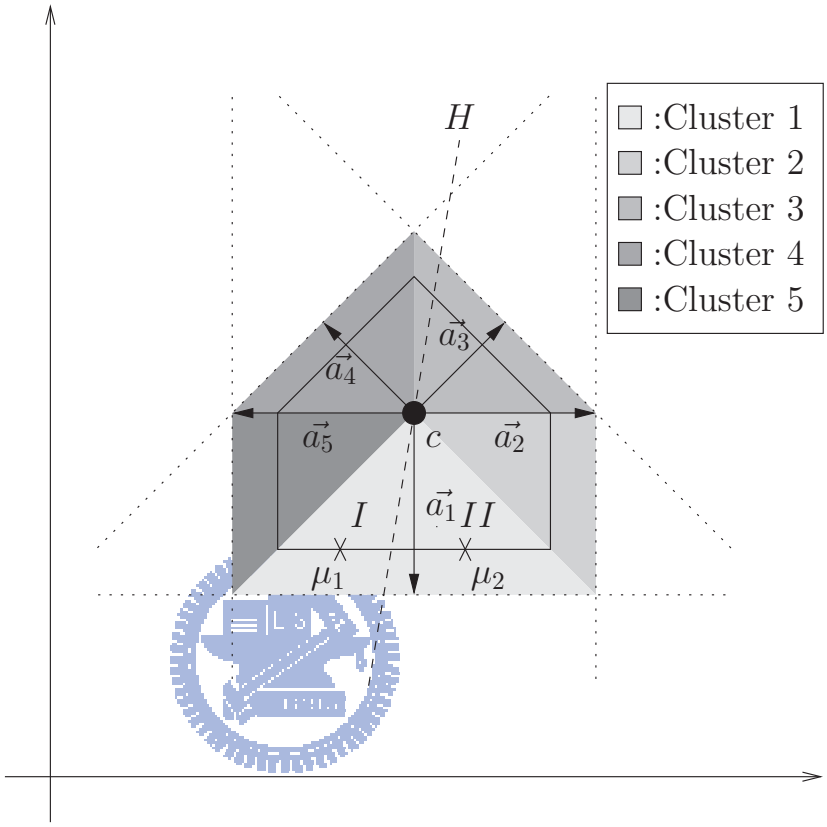


Figure III.11: An example to show the partitions of data cluster and the normal vector orientation estimation.

2.3 Self-Growing Learning Algorithm

To decide the number of normal vectors of a polygon descriptor, a systematic process is proposed in this Section to gradually add/remove normal vectors until the number of normal vectors converges. Since side clusters which share the same side boundary are associated to identical normal vectors, these side clusters can be merged by removing identical normal vectors. That is, the number of normal vectors will converge to the number of pseudo boundaries when gradually increasing the number of normal vectors.

The flow chart of the proposed self-growing learning process for Polygon descriptor is shown in Figure III.12. At first, Medoid is subtracted from every data points. That's, the data distribution is translated to let its Medoid located at Origin. Then, normal vectors are initialized in a random manner. According to the normal vectors, data points are clustered into side clusters. For each side cluster, a normal vector is estimated according to the data distribution of data points in the side cluster. Then, for side clusters with identical normal vectors, their normal vectors are merged and the data points are marked as frozen. For the rest of side clusters, their normal vectors will be used by another round of clustering and estimating processes until all data points are marked as frozen.

When implementation, identical normal vectors usually means that the angle between two normal vectors is smaller than a threshold, because there may exist precision or rounding error. Such approximation let a polygon descriptor doesn't need to use infinite boundaries to describe a curve. However, a data distribution may has more than one possible polygon descriptor because errors may be approximated in different way.

As shown as the exemplar learning process in Figure III.13, a set of data points which uniformly distributed in a triangular region is given. At first,

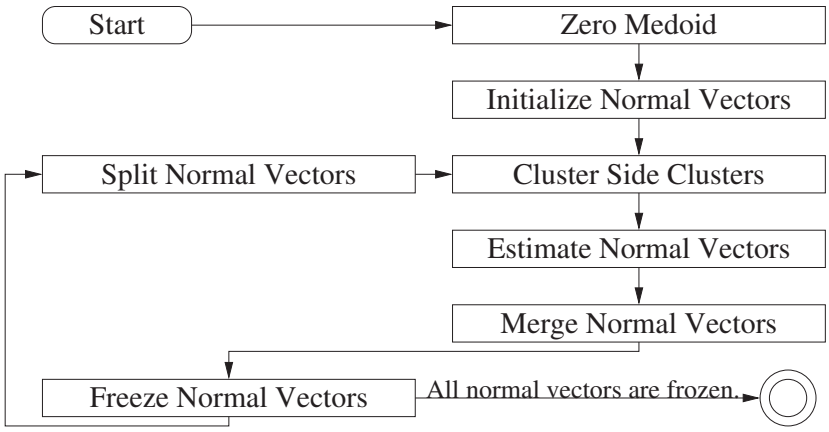


Figure III.12: The flow chart of self-growing learning process for Polygon descriptor. By iteratively splitting, clustering, estimating, and merging normal vectors, the number of normal vectors can be decided automatically, when learning the shape of data distribution.

two normal vectors are initialized. And, the data points are clustered into two side clusters according to the normal vectors. Then, the normal vectors are re-estimated according to the data points in each side clusters. By iteratively applying the clustering, estimating, and splitting process several times, 3 pairs of identical normal vectors are found (the third row). By merging these identical normal vectors, data points located in the dark-gray area are marked as frozen. Such processes will be repeatedly invoked until most of the data points are marked as frozen.

Steps of freezing data points, is essential to ensure that the learning (fitting) will converge. However, freezing data points decreases the number of available sample point for mean computing. To increase the robustness of normal vector estimation, using related frozen data points to support mean computation could minimize the damage of freezing data points in implementation.

Although the proposed method try to fit the distribution shape as pre-

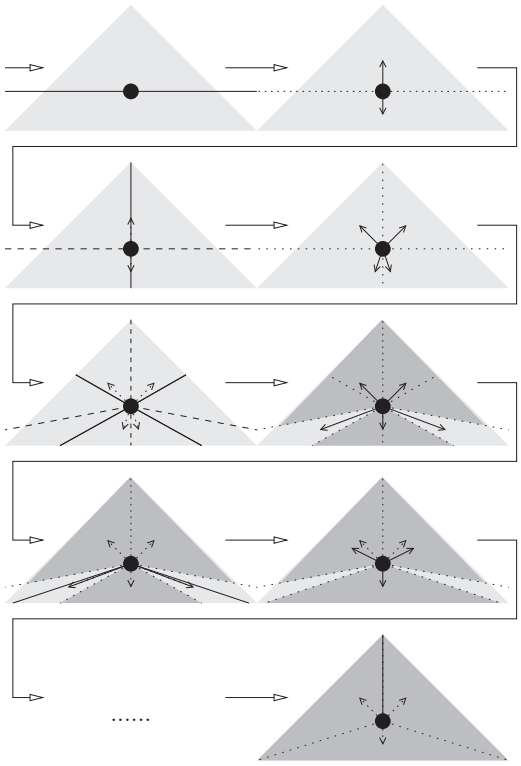


Figure III.13: An exemplar learning process of Polygon descriptor. At first, two normal vectors are initialized. When the number of normal vectors increased to 8 (third row), 3 pairs of axes are merged and the normal vectors points associated to the side clusters in dark-gray area are marked as frozen (3th row, 2nd col). Such processes will be repeatedly invoked until most of the data points are marked as frozen.

cisely as possible, the proposed method may find more than one possible polygon descriptor with random based initialization for some situation. For example, if data distributed within a circular region, changing initial conditions could make the estimated polygon descriptor rotate, because a polygon with infinite boundaries is impractical and the fitting method has to allow a tolerance to approximate the circular distribution by a polygon with finite boundaries. Figure III.14 shows an example about the alignment problem of polygon description estimation. When approximating a smooth circle by

finite number of boundaries, there are more than one result can achieve the best precision. That's, how to align the estimation polygon descriptors is an important issue of extensions of real world application.

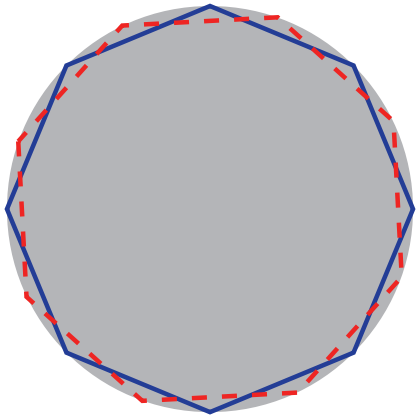


Figure III.14: More than one polygon descriptors model the same circular data distribution. If a polygon can have infinite boundaries, then these polygon descriptors will be identical. However, we have to approximate a circular distribution by a polygon with finite boundaries. And rotating an estimated polygon descriptor can have another one with the same precision, because the real data in distributed in a circular region.

3 Evaluation and Experimental Results

An exemplar Polygon descriptor estimated from a data set sampled from Taiwan Stock Market is shown in Figure III.15. According to the price and earn per share (EPS) of stocks collected in February 2003, a data distribution is drawn. The data distribution is normalized and translated to let the Medoid of data point located at origin. By using the learning algorithm introduced in Section 2.3, four normal vectors and four side clusters are estimated. Data points in different side clusters are painted in different colors and the pseudo boundaries of each side clusters are drawn in bold lines.

Synthesis data are generated by random number generator with various random seeds to test the performance of learning algorithm. Figure III.16 shows how the number of data points affects the precision of learning algorithm. The precision of learning algorithm is represented by the cosine value $\cos(\delta)$ of the angle between estimated normal vectors and ground truth. More data points makes the synthesis data distribution approximates the uniform distribution better. Thus, the estimated normal vectors match the ground truth better. The horizontal axis is for the number of data points and the vertical axis is for the precision. As the results shown in Figure III.16, the precision rate converges when data points are more than 1000. Therefore, the number of synthesis data points for experiments in this dissertation is set to 3000.

Figure III.17 shows the efficiency of learning algorithm. 12000 random generated data points are located uniformly in a diamond-shape region. At the beginning, four axes are initialized as $(1,0)$,$(0,1)$,$(-1,0)$, and $(0,-1)$. The precision of estimated normal vectors are represented by the average of four cosine value between the estimated normal vectors and the ground

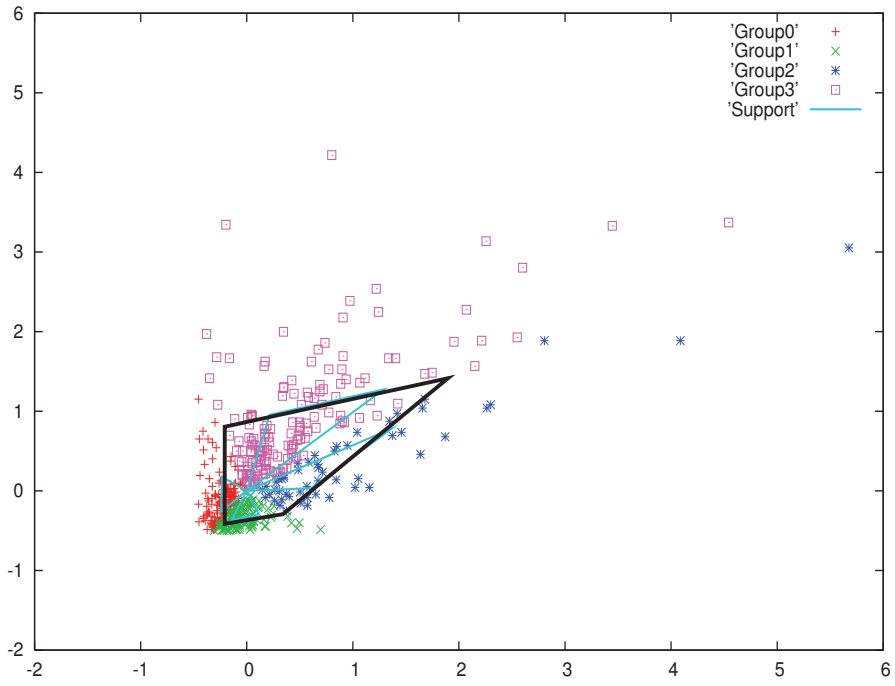


Figure III.15: An exemplar Polygon descriptor estimated from a data set collected from Taiwan Stock Market. The horizontal axis is for stock price and the vertical axis is for EPS (earn per share). According to the price and EPS of a stock, a data point is drawn in the data distribution. The data distribution is normalized and translated to let the Medoid of data point located at origin. Four normal vectors are found, and the corresponding pseudo boundaries are drawn in bold lines. Data points in different side clusters are painted in different colors.

truth. As the results shown in Figure III.17, the estimated normal vectors converges to ground truth in less than 10 iterations.

Generally speaking, wider a data distribution along a pseudo boundary, more precise the estimation of normal vector orientation can achieve. For 2-D data, the vertex angle, located at reference center, of a side cluster can be used to represent how width a data set distributes along a pseudo boundary. Figure III.18 shows how the vertex angle of a side cluster affects the precision of normal vectors orientation estimation. The horizontal axis is for the angel

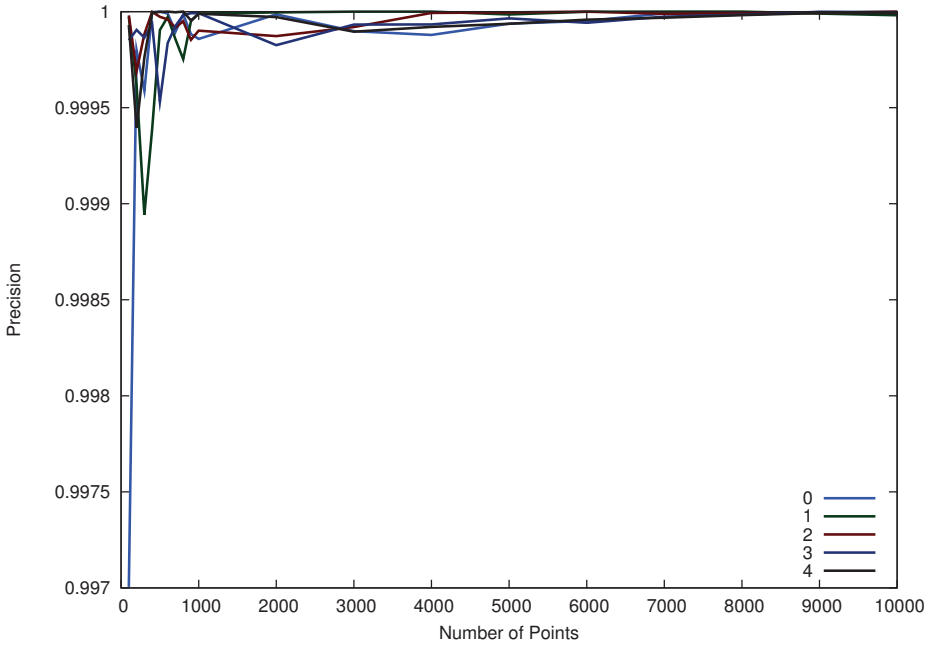


Figure III.16: Number of data points V.S Precision of learning algorithm. The horizontal axis is for the number of data points and the vertical axis is for the precision of normal vectors estimation. More data points makes the synthesis data distribution approximates the uniform distribution better. Therefore, the estimated orientation of normal vectors match the ground truth better.

of a side cluster and the vertical axis is for the precision of learning algorithm. As the results shown, when the vertex angle is smaller than 15 degree, the precision of normal vector estimation is decreased.

Since the proposed method merges similar normal vectors in learning process, when the angle between two pseudo boundaries is very small, the corresponding side clusters/normal vectors may be merged by mistake. Figure III.19 shows how the angle between two pseudo boundaries affects the precision of learning algorithm. The horizontal axis is for the angle between two neighboring pseudo boundaries and the vertical axis is for the precision of learning algorithm. According to the experimental results, the learn-

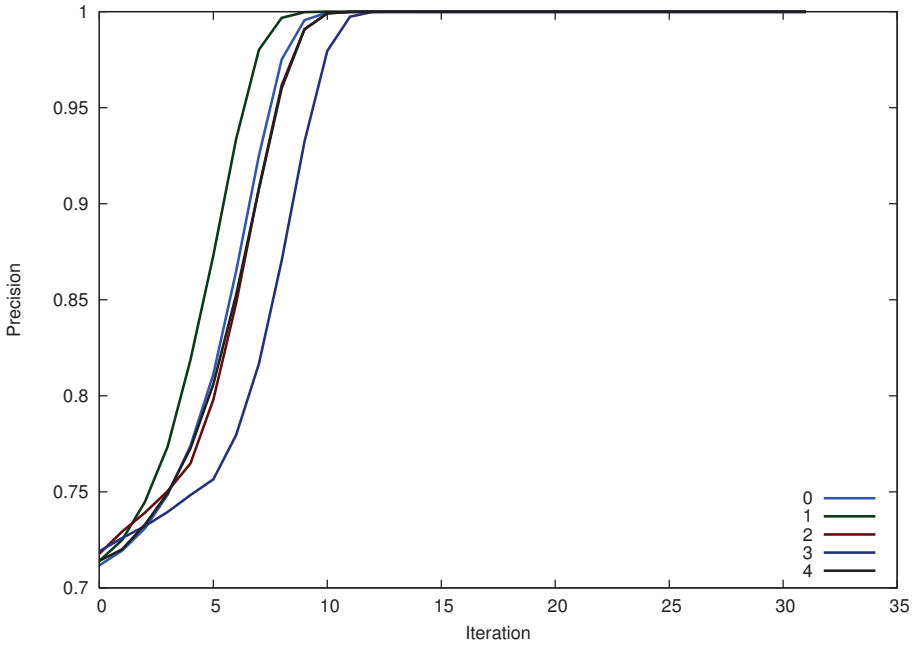


Figure III.17: Learning Iteration V.S Precision of learning algorithm. The horizontal axis is for the number of learning iterations and the vertical axis is for the precision of normal vectors estimation. In this experiments, the learning algorithm converges in less than 10 iteration.

ing algorithm merges two side cluster when the angle between their pseudo boundaries is smaller than 11 degree (0.98 in cosine value).

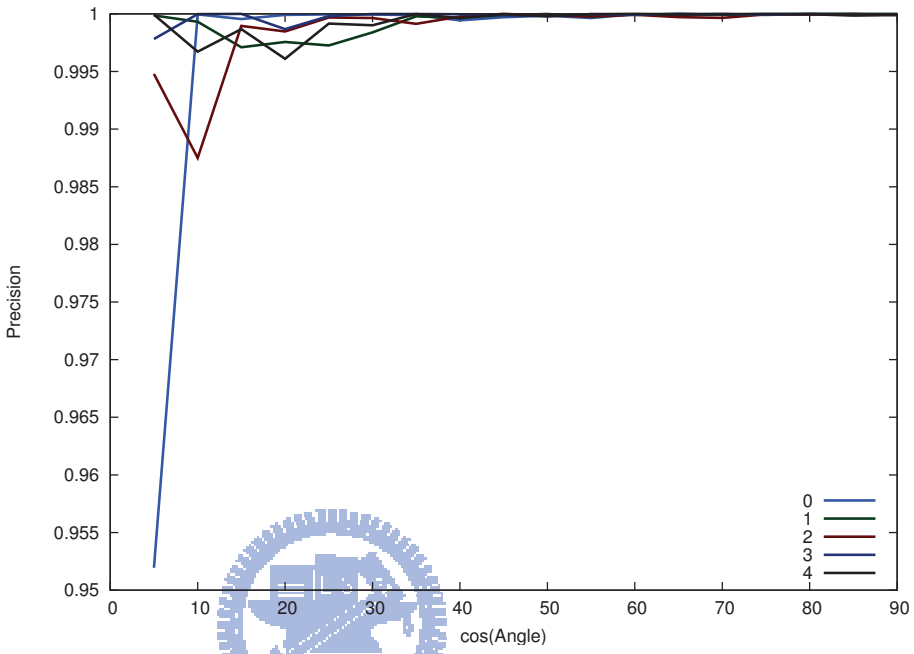


Figure III.18: Angle of a side cluster V.S Precision of learning algorithm. The angle of a side cluster reflects how width a data set distributed along the pseudo boundaries. Generally, wider a data distribution along a pseudo boundary, more precise the estimation of normal vector orientation can achieve.

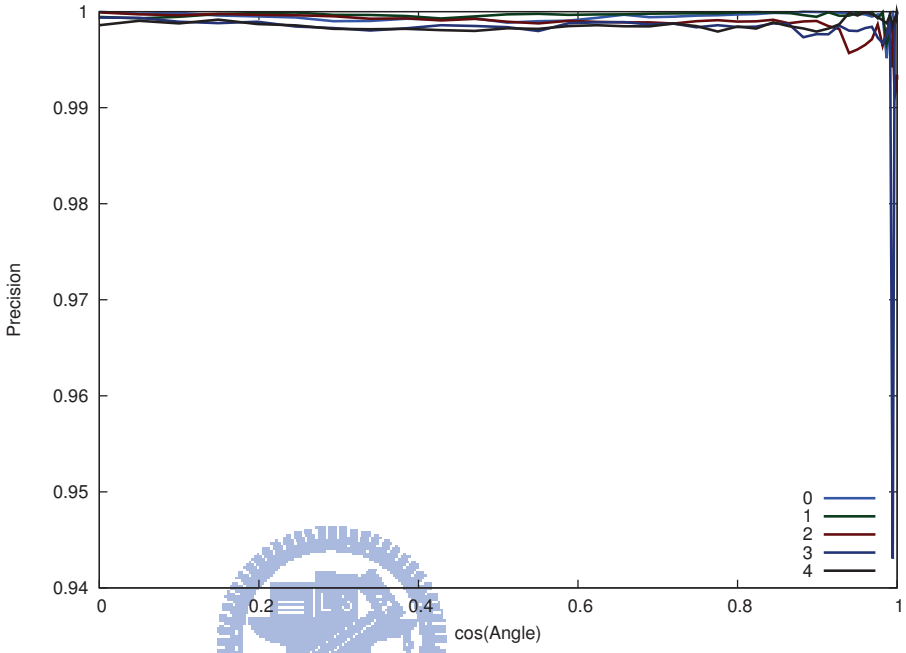


Figure III.19: Angle between two pseudo boundaries V.S Precision of learning algorithm. Since the proposed method merges similar normal vectors in learning process, when the angel between two pseudo boundaries is small, the corresponding side clusters are merged by mistake. Thus, the precision of learning algorithm decrease when the angle between two pseudo boundaries decrease.

4 Variant Polygon Descriptors

This section is beyond the scope of this dissertation. That is, information in this section is not essential for the later materials of this dissertation. Because I mentioned that Polygon descriptor is potentially to have a variant form which works for clustering problem, I am going to show some possible designs for such a clustering variant. However, since this dissertation is not going create any application based on these variants, the methods proposed in this section only suggest the idea of what's useful when designing extensions.

Two variants are suggested to be the variants of polygon description in clustering - 1) **Joint Polygon Descriptor** and 2) **Composite Polygon Descriptor**. **Joint Polygon Descriptor** represents unconnected data distributions with multiple polygon descriptors. On the other hand, **Composite Polygon Descriptor** is used to separate a composite data distribution to several **regular components**, which will be defined in Figure III.20. As shown in Figure III.20, a regular component is defined as data distribution which is composed by various uniform regions in the same shape and sharing the same center.

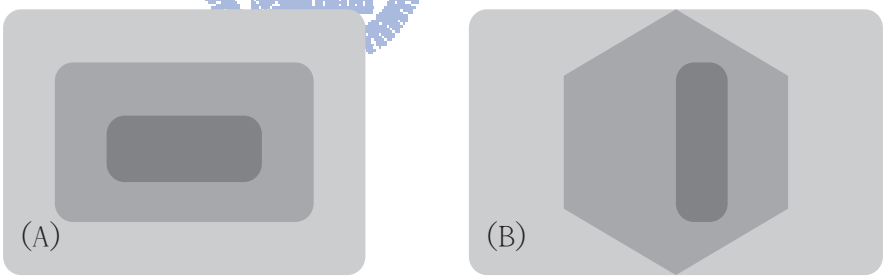


Figure III.20: An example of equivalent component for Composite Polygon Descriptor. (A) Since the regions of three densities are in the same shape and share the same center, they are treated as a single regular component. (B) Since each region with uniform densities are in different shape or having different center, they are treated as three different regular components.

4.1 Joint Polygon Descriptors

Joint Polygon Descriptor try to separate unconnected region to multiple polygon descriptors. For example, two unconnected region in Figure III.21 are supposed to be represented by two polygon descriptors. Without using multiple polygon descriptors, the contour of estimated polygon descriptor will looks like a (the blue lines). Based on the shape a, data samples can be weighted according to the projection ratio on each normal vectors. Then, the shape b (the red lines) can be estimated based on the weighted data samples. By repeating such process, the exact shape of left rectangle c will be available.

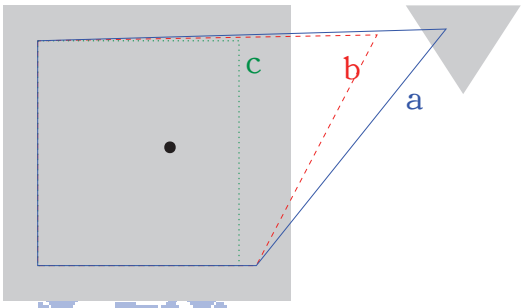


Figure III.21: By weighting data samples according to the projection ratio on each normal vectors, the estimated polygon descriptor changed from a,b, to c.

Joint Polygon Descriptor relies on the following property: " *When removing data samples in outer region by resizing of Polygon descriptor without changing its shape, the estimated shape will remain the same if the data distribution is composed by uniform distribution regions in the same shape and sharing the same center.*" Based on the polygon probability density function shown in Figure III.1 (b), data samples can be weighted according to the projected distance from reference center. And the job of removing data samples in outer bands can be done by simply adjusting the variance in each

orientation of normal vectors. The estimation process could be designed as follows,

step 1. Let weights of all data samples in the first polygon descriptor equal to 1.

step 2. Estimate polygon descriptors based on the weighted sample data distribution.

step 3. If the estimation didn't changed the shape of polygon descriptor, goto **step 6.**.

step 4. Re-weight data samples by polygon probability density function(see Figure III.1 (b)).

step 5. Goto **step 2.**.

step 6. Compute the complement of weights by subtracting the weights of existing polygon descriptors.

step 7. If the complement of weights almost equal to zero, goto **step 10.**.

step 8. Initialize a new polygon descriptor with the weight of data samples equal to the complement of existing weights.

step 9. Goto **step 2.**.

step 10. **[DONE]**

4.2 Composite Polygon Descriptors

Composite Polygon Descriptor separates a composite data distribution to several **regular components**, which is defined in Figure III.20. Unlike Joint Polygon Descriptor, the weight of data samples can not be extracted directly because the components are overlapped. The overlapped ratio should be tested gradually until various shape regions are extracted one by one.

Figure III.22 shows an example data distribution which requires Composite Polygon Descriptor to separate components. The inner component, dark gray diamond area, can be located by repeatedly estimating Polygon Descriptor and Removing data samples in outer band, like Joint Polygon Descriptor does.

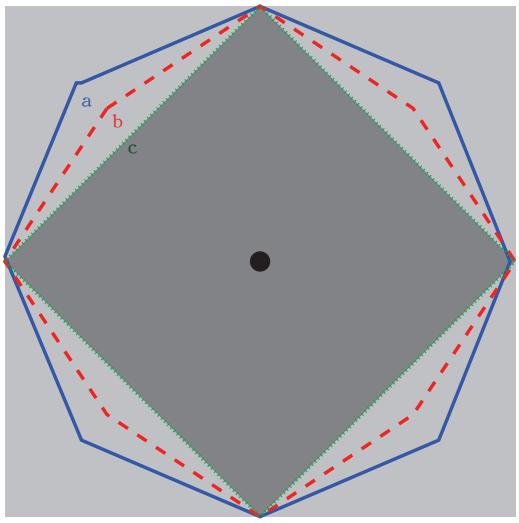


Figure III.22: A data distribution composed by two uniform distributed regions. The diamond shaped region is overlapped above the rectangle region, and the diamond one is completed contained by the rectangle one. By changing the weight of data samples, the estimated polygon description (eg. blue, red, green contour) changed.

After locating the centered small region, the estimation for larger regions need to find a weight to remove the centered small region. The weight

σ should be tested gradually. That is, as shown as Figure III.23, various weighting curve are applied and validated to find which one can correctly remove the affect from inner small region.

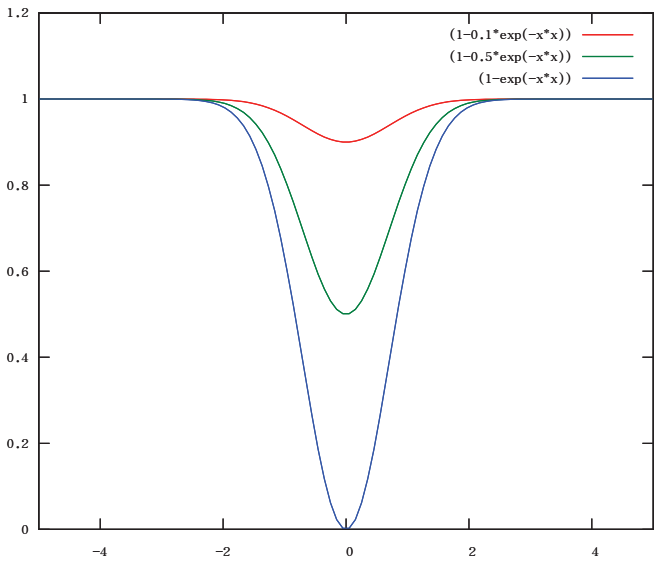


Figure III.23: The curve is the complement of a weighted Gaussian function. $1 - \sigma e^{x^2}$

Composite Polygon Descriptor could be designed as follows,

step 1. Let weights of all data samples in the first polygon descriptor equal to 1. σ, the overall weight of last component, equal to 0.

step 2. Estimate polygon descriptors based on the weighted sample data distribution.

step 3. If the estimation didn't changed the shape of polygon descriptor, goto **step 5.**.

step 4. Shorten the length of each normal by a small fixed ratio, updated the weight of data samples, and goto **step 2.**.

step 5. If the estimated polygon descriptor didn't similarity to the last component, goto **step 8.**.

step 6. Increase σ, and apply the weighted probability function of complement polygon descriptor,$p'(x) = 1 - (\sigma p(x))$, to gradually remove the data samples in the last component.

step 7. Goto **step 2.**.

step 8. Report last component, extract it, and reset σ to 0.

step 9. If all data samples are located inside the region of last component, goto **step 11.**.

step 10. Goto **step 2.**.

step 11. [**DONE**]

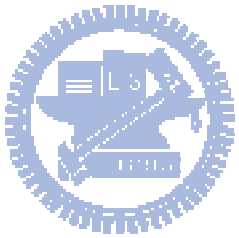
5 Concluding Remarks

Polygon descriptor is designed to be a numerical representation of a data distribution. This representation should be suitable for data samples in any dimension. By describing a high dimensional data as polygon descriptors, we can analyze the characteristics of a high dimension data distribution. By applying some user interfaces design, we can even visualize the high dimensional data distribution. However, most of such applications depending on the requirements of the related real world application.

Discovering the trend of data distribution is one motivation of designing Polygon Descriptor. Therefore, the estimation method depending on the global statistic information instead of the boundaries on its contour. However, the other estimation methods can be designed to achieve different target. For example, Since multi-layer perceptron can be used to learn the boundaries of a data distribution, the normal vector could be decided by perceptrons too. Based on different motivation, different estimation methods can be created to let the estimated polygon in desired shape.

In this dissertation, three Polygon descriptor based methods, 1) measurement of shape deformationIV, 2) virtual geometry for similarity based clusteringV, and 3) polygon-based region representationVI, are proposed. Measurement of shape deformation estimates the similarity of data distributions. The proposed measurement method computes the similarity of data sample groups collected from Taiwan stock market. Since the shape of data distribution implies the system behavior of a system, the behavior in periods of stock market can therefore be clustered. Then, virtual geometry is proposed to emulate the geometry of a virtual feature space. That is, for similarity based clustering, such as K-medoid, there is no geometry informa-

tion about data points. To estimate variances for these clustering models, a virtual geometry is required. Last, Since Polygon descriptor measures the shape and coverage area of a data distribution, the region selection and region tracking methods can also be developed based on Polygon descriptor.



Chapter IV

Measurement of Shape Deformation

Based on Polygon descriptor, a data distribution can be represented by a numerical/model form. By using the numerical form of a polygon descriptor, a value can be measured to represent the similarity between two data distributions. In Section 1, general methods that measure the similarity between two numerical/model representation are introduced at first. Then the similarity measurement method, named Deforming distance, is proposed in Section 2 to measure the similarity value between two polygon descriptors. Since the deforming distance is difficult to handle high dimension data, Minimal-cost normal vectors matching is then proposed in Section 3 to measure the similarity between polygon descriptors in high dimension space. A real world application, which shows the similarity between every two periods of Taiwan Stock Market, is demonstrated in Section 4. Last, the concluding remarks are drawn in Section 5.

1 Related Works

1.1 Correlation Coefficient

Beside extracting statistical characteristic from a data distribution, Correlation coefficients can also be used to compare the similarity between two number sequences. Given two number sequences $H_A = \{a_1, ..., a_N\}$ and $H_B = \{b_1, ..., b_N\}$, μ_A and μ_B are the means of H_A and H_B respectively. Covariance can be defined as follows,

$$K_{CV} = \frac{1}{N}\sum_{i=1}^{N}\{(a_i - \mu_A)(b_i - \mu_B)\} \tag{IV.1}$$

By normalizing the data sequences with standard derivations σ_A and σ_B, correlation coefficient can be defined as

$$K_{CR} = \frac{K_{CV}}{\sigma_A \sigma_B} \tag{IV.2}$$

Let H'_B be another sequence generated by translating and scaling H_B. Since

$$\begin{aligned}
\mu_{H'_B} &= \frac{\alpha}{m}\sum_{i=1}^{m} b_i + \beta = \alpha\mu_{H_B} + \beta \\
\sigma_{H'_B} &= \frac{\alpha}{m}\sqrt{\sum_{i=1}^{m} b_i^2} = \alpha\sigma_{H_B} \\
CV_{H'} &= \frac{\alpha}{m}\sum_{i=1}^{m}\{(a_i - \mu_{H_A})(b_i - \mu_{H_B})\} = \alpha CV_H \\
CR_{H'} &= \frac{1}{m\sigma_{H_A}\sigma_{H_B}}\sum_{i=1}^{m}\{(a_i - \mu_{H_A})(b_i - \mu_{H_B})\} \\
&= CR_H,
\end{aligned}$$

therefore, correlation coefficient is invariant to translate and scale.

1.2 K-L Divergence

When a data distribution is represented by continuous data models, such as Gaussian Mixture model, K-L divergence can be used for similarity measure-

ment. K-L divergence, also called relative entropy, is defined as follows

$$D_{KL}(p\|q) = \int p(x) \log \frac{p(x)}{q(x)} dx \tag{IV.3}$$

where $p(x)$ and $q(x)$ are two probability distributions. K-L divergence is widely used as a quantity which measures the difference from a true probability distribution $p(x)$ to an arbitrary probability distribution $q(x)$.

Equation IV.3 can be expanded as follows

$$D_{KL}(p\|q) = \int p(x) \log p(x) - \int p(x) \log q(x) \tag{IV.4}$$

The first term is the entropy of $p(x)$. The entropy of $p(x)$ is the lower bound of average bit-length to encode data with probability distribution $p(x)$. The second term represents the average bit-length to encode data with probability distribution $p(x)$ with the optimal encoding scheme for probability distribution $q(x)$. When $p(x)$ is equal to $q(x)$, K-L divergence is zero. Since K-L divergence is not symmetric, $D_{KL}(p\|q) + D_{KL}(q\|p)$ is generally used to represent the similarity between two models.

Although mixing multiple Gaussian components makes a data model capable to approximate all kinds of distribution shapes, integration operation is difficult to implement for mixing model. That is, there is no close-form expression for the KL-divergence between two mixtures of Gaussian. Therefore, Monte-Carlo simulation is used to approximate the KL-divergence between two Gaussian Mixtures p and q:

$$D_{KL}(p\|q) = \int f \log \frac{p}{q} \approx \frac{1}{n} \sum_{i=1}^{n} \frac{p(x_i)}{q(x_i)} \tag{IV.5}$$

where x_i is sampled from $f(x)$.

1.3 Chi Square(χ^2) Test

χ^2 test is generally used to test the fitness of distribution assumptions for statistical applications. Giving a distribution function $p(x)$ and a set of data

points $X = \{x_1, ..., x_N\}$, χ^2 test measures the fitness of a given probability density function $p(x)$ and a point set X.

Fukunaga et al.[8] discuss several related applications of optimum error-reject functions. Based on several thresholds of probability values, a space is divided into regions. If the distribution of data points fits the probability density function, the ratio of points belong to X located in each region should be similar to the ratio of distributed area of $p(x)$ in each regions.

Giving k thresholds $(t_1 < ... < t_k)$, the space is partitioned into k regions. χ^2 test is defined as follows

$$Q = \sum_{i=1}^{k} \left[\frac{(N[R_{i-1} - R_i] - N[r_{i-1} - r_i])^2}{N[r_{i-1} - r_i]} \right] \tag{IV.6}$$

where N is the total number of points in X, k is the number of testing regions, $r_i = P(p(x) > t_i)$, and

$$R_i = \frac{1}{N}(\text{ number of } p(x_1), p(x_2), ..., p(x_N) > t_i)$$

Figure IV.1 gives an example of χ^2 test. In this example, $k = 2$, $N = 8$, $R_1 = 8/8 = 1$, $R_2 = 6/8 = 0.75$, $r_1 = \frac{2wh}{2wh} = 1$, $r_2 = \frac{1.5wh}{2wh} = 0.75$, and $Q = \frac{[8(1-0.75)-8(1-0.75)]^2}{8(1-0.75)} = 0$.

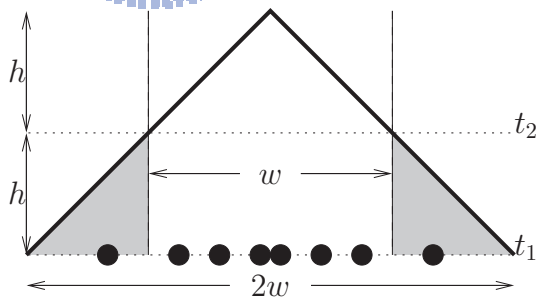


Figure IV.1: An example of χ^2 test. In this example, $k = 2$, $N = 8$, $R_1 = 8/8 = 1$, $R_2 = 6/8 = 0.75$, $r_1 = \frac{2wh}{2wh} = 1$, $r_2 = \frac{1.5wh}{2wh} = 0.75$, and $Q = \frac{[8(1-0.75)-8(1-0.75)]^2}{8(1-0.75)} = 0$.

The similarity value between two random sets can therefore be estimated by cross validating a data set and a statistical model using χ^2 test.

1.4 Hausdorff Distance

Hausdorff distance is used to measure the difference between two data sets. Hausdorff metric couples elements in one set and the other depending on their distance. That is, for each data element in one data set, the smallest distance to a data point in the other set is measured to represent the *point-to-set* distance. The largest *point-to-set* distance is used to represent the distance between two data sets.

Hausdorff distance between two sets A and B, is defined as

$$H(A,B) = max\left(h(A,B), h(B,A)\right), \tag{IV.7}$$

where

$$h(A,B) = \max_{a\in A} \min_{b\in B} \|a-b\|,$$

and $\|\cdot\|$ is some underlying norm of A and B. $h(A,B)$ is called the direct Hausdorff distance from A to B.

An example of Hausdorff distance measurement is shown in Figure IV.2. The Hausdorff distance from set A to B is measured from an element in A that has the *largest minimal distance* to an element in B. Obviously, $h(A,B)$ may not equal to $h(B,A)$. In order to have a symmetric distance measurement, the Hausdorff distance between A and B is defined to be the larger one of $h(A,B)$ and $h(B,A)$.

Huttenlocher et al.[13] use Hausdorff metric to measure the similarity between images. Since the Hausdorff distance computation differs from many other image comparison methods in that no correspondence between the

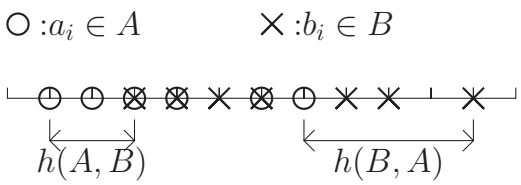
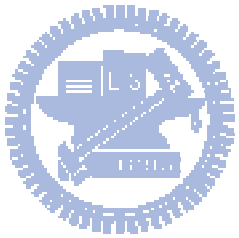


Figure IV.2: An example of Hausdorff distance measurement. The largest minimal distance from an element in set A to an element in set B is measured to represent the similarity value between two sets A and B. In this example, $h(A, B)$ is 2, $h(B, A)$ is 4, and the Hausdorff distance $H(A, B)$ is 4.

model and the image is derived. The method is quite tolerant of small position errors such as those that occur with edge detectors and other feature extraction methods. However, since the Hausdorff distance may involve computation of largest minimal distance from outlier points, the Hausdorff distance measurement can be very sensitive to outliers.



2 Deforming Distance

Deforming distance is proposed to represent the difference between polygon descriptors. Given a polygon descriptor with N normal vectors $\{n_0, ..., n_{N-1}\}$, a number sequence, $\{a_0, a_1, ..., a_{N-1}\}$, can be used to represent the Polygon descriptor. The number sequence $\{a_0, a_1, ..., a_{N-1}\}$ can be enumerated as follows:

$$a_i = \|a_i'\| = \left\| cos^{-1}\left(\frac{n_i \cdot n_{(i+1)\%N}}{\|n_i\|\|n_{(i+1)\%N}\|}\right)\right\|, \text{ for } i = 0, ..., N-1,$$

where a_i' is the angle between two adjacent normal vectors n_i and n_{i+1}, a_i is the magnitude of a_i' and "$\%N$" is the modular N operation. Given a polygon descriptor, the shape of corresponding polygon is encoded by the number sequence $\{a_0, a_1, ..., a_{N-1}\}$, since the angles between any two adjacent boundary edges of the corresponding polygon is equal to $\pi - a_i'$ where n_i and n_{i+1} are the corresponding normal vectors of the boundary edges.

The problem of measuring the difference between two polygon descriptors are reduced to the problem of measuring the difference between two number sequences. The Deforming distance defined in Section 2.1 is used to measure the difference between two number sequences. The methods that estimating the Deforming Distance are then introduced in Section 2.2.

2.1 The Definition of Deforming Distance

Let $\{a_0, ..., a_{N-1}\}$ be a number sequence, three deforming operations are defined as follows:

1. **Deletion**: $delete(i)$ - if $a_i = 0$, remove a_i from the number sequence.

2. **Insertion**: $insert(i)$ - insert a zero in the number sequence next to a_i.

3. **Chang**: $change(i, \delta)$ - add 2δ to a_i and subtract one δ from both $a_{(i-1)\%N}$ and $a_{(i+1)\%N}$.

Given two number sequences $S = \{s_0, ..., s_n\}$ and $T = \{t_0, ..., t_M\}$, a sequence of deforming operations can change S to T. The cost of the sequence of deforming operations is defined as the sum of change amount δ for all the needed *Change* operators. *Insertion* and *Deletion* operations require zero cost of the sequence of deforming operations, because these two operations just insert or remove a zero to or from number sequence. The minimal cost to convert a number sequence S to T using deforming operations is defined as the Deforming distance. And, the corresponding sequence of deforming operations for the Deforming distance is called Deforming path.

Based on the three deforming operations: 1) deletion, 2) insertion, and 3) change, the deforming distance is invariant to translation, rotation, and scaling. By defining more specific deforming operations, the invariance of similarity measurement, deforming distance, is changed. By appending two extra deforming operations: 4) rotation, and 5) extension, the measuring of deforming distance can do exact match between two polygon descriptors.

The functions of five complete deforming operations can de illustrated by Figure IV.3. As the figure shown, a *change* sharpen/widen a side cluster of a polygon descriptor. A *deletion* removes an existing side cluster from a polygon descriptor and an *insertion* appends a new side cluster into a polygon descriptor. A *rotation* rotates a polygon descriptor and an *extension* stretch normal vectors of a polygon descriptor. Since Deforming distance is required to do translation, rotation and scaling invariant similarity measurement for this dissertation, only the first three deforming operations are discuss in the section. As the example shown in Figure IV.4, by applying a series of deforming operations, a triangle can be transformed into a diamond. By

estimating the minimal transformation cost to transform a polygon descriptor to another, the similarity between them can therefore be measured.

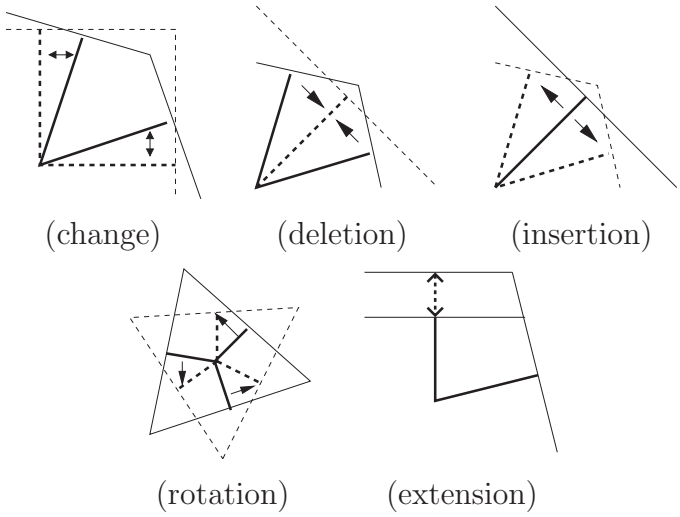


Figure IV.3: The physical meaning of all five deforming operations are shown. A change sharpen/widen a side cluster. A deletion/insertion removes/appends a side cluster. A rotation rotate a polygon descriptor, and a extension stretch a normal vector of a polygon descriptor. There five deforming operations are for 2-D shape only. Polygon descriptors in higher dimension requires more operations to from a transforming process, because the angle between normal vectors is a high dimension space angle.

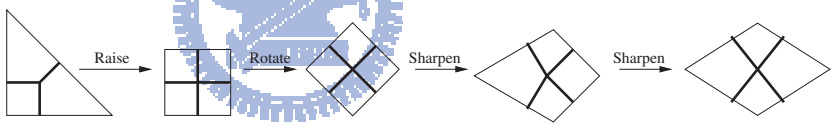


Figure IV.4: By applying a series of deforming operations, a triangle can be transformed into a diamond.

In considering that a polygon descriptor represents the distribution shape of data samples, each deforming operation implies one kind of system behavior change. For example, applying a changing operator to decrease an angle between two normal vectors will results the shape sharper. Usually, a sharp angle means that the data distribution on the orientation sticky to a line. Thus, related factors may getting more independent.

2.2 The Estimation of Deforming Distance

Many well-known algorithms, such as shortest-path[21][3], traveling salesman problem[22], string-to-string correction problems[23][28], are proposed to find the optimal path for *path-finding* problems. Among these algorithms, string-to-string correction problem is very similar to the problem of finding the Deforming path and Deforming distance described in Section 2.1. By referencing the correction path of the string-to-string correction problem, match assignments are proposed to divide the problem of Deforming path finding to several smaller sub-problems. Then, the Deforming distance for the sub-problems can be estimated.

A match assignment describes the relationship of elements between source and target number sequence. Let $S = (s_0, ..., s_{n-1})$ and $T = (t_0, ..., t_{m-1})$ be source and target sequences. Let a match assignment between S and T be $M = \{(l_0^s, l_0^t), ..., (l_{k-1}^s, l_{k-1}^t)\}$, where k is the number of entries in M, $0 \leq l_i^s \leq n-1$, $0 \leq l_i^t \leq m-1$, $0 \leq i < k-1$, and l_i^s and l_i^t are the labels of the i-th number in S and T respectively. An element (l_i^s, l_i^t) in a match assignment means that $t_{l_i^t}$ in T is associated with $s_{l_i^s}$ in S. For example, the match assignment {(0,1),(1,1),(1,2),(2,0)} means that s_0 in S is associated with t_1 in T, s_1 in S is associated to t_2 and t_3, in T, and s_2 in S is associated to t_0 in T.

When the relationship between the elements of source sequence and the elements of target sequence is fixed, the minimal cost of deforming operations to convert the source sequence to the target sequence can be measured by the methods presented in Section 2.2. By computing the minimal cost of deforming operations for every possible relationship between the elements of source sequence and the elements of target sequences, the deforming distance which is the minimal cost of deforming operations without constraining the

relationship of the elements between source sequence and the elements of target sequence, is available. The method that generates match assignments for every possible relationship between the elements of source sequence and the elements of target sequence is illustrated in Section 2.2.

Local Estimating of Deforming Distance

Given a match assignment M between two number sequences S and T, a number of zeros are inserted into S and T to equalize the number of elements in these two sequences. Two modified source and target sequences S' and T' of equal number of elements are generated as follows:

- For two entries (l_i^s, l_i^t) and (l_{i+1}^s, l_{i+1}^t) in M, if $l_i^t = l_{i+1}^t$, then a zero is inserted right after $t_{l_i^t}$.
- For two entries (l_i^s, l_i^t) and (l_{i+1}^s, l_{i+1}^t) in M, if $l_i^s = l_{i+1}^s$, then a zero is inserted right after $s_{l_i^s}$.

Both of the modified sequences $S' = \{s'_0, ..., s'_{N'-1}\}$ and $T' = \{t'_0, ..., t'_{N'-1}\}$ contain N' elements. By transferring a partial amount of value from elements in S' to the elements next to them, S' can be converted into T', where transferring a partial amount of value d_i from s'_i to s'_{i+1} means that d_i are subtracted from s'_i and added to s'_{i+1}.

A sequence $D = \{d_0, ..., d_{N'-1}\}$, called *Shift amount*, indicates the minimal transferring values d_i needed to convert S' to T'. Using the match assignment M between $S' = \{s'_0, ..., s'_{N'-1}\}$ and $T' = \{t'_0, ..., t'_{N'-1}\}$, *Shift amount* can be evaluated as follows:

1. First, let us map S' and T' to a coordinate system (x, y), as shown in Figure IV.5.

2. Then, plot N' points $\{m_0, ..., m_{N'-1}\}$ where the point m_i is located at $(\sum_{j=0}^{i} t'_j, \sum_{j=0}^{i} s'_j)$, for $i = 0, ..., N'-1$. In Figure IV.5, the points $\{m_0, ..., m_{N'-1}\}$ are marked in solid black dots.

3. Draw a line $y = x + \beta$, and plot another N' points $\{p_0, ..., p_{N'-1}\}$ which are one-to-one corresponds to $\{m_0, ..., m_{N'-1}\}$ and each point p_i is located at $(\sum_{j=0}^{i} t'_j, \beta + \sum_{j=0}^{i} t'_j)$, for $i = 0, ..., N'-1$. In Figure IV.5, the points $\{p_0, ..., p_{N'-1}\}$ are drawn in solid squares.

4. Change the value of β, such that $\sum_{i=0}^{N'-1}(x_i^m - x_i^p)$ equals to zero where (x_i^m, y_i^m) and (x_i^p, y_i^p) are the coordinate of m_i and p_i respectively.

5. Calculate shift amount d_i in D. For each point m_i, the shift amount d_i is equal to $y_i^m - y_i^p$. Thus, the Shift amount d_i can be rewritten as follows:

$$d_i = \left(\sum_{j=0}^{i} s'_j\right) - \left(\beta + \sum_{j=0}^{i} t'_j)\right), \text{ for } i = 0, ..., N'-1.$$

The shift amounts d_i can be solved by finding β which minimizes $\sum_{i=0}^{N'-1} \|d_i\|$.

Since the deforming distance is defined as the sum of deforming operations, the shift amounts d_i have to be converted to δ_i which is used by the *changing operations*. An example shown in Figure IV.6 illustrates the relation between δ_i and d_i. In general, the relation between δ_i and d_i are summarized as follows,

$$d_i = \delta_i - \delta_{(i+1)\%N'}, \text{ for } i = 0, 1, ..., N'-1.$$

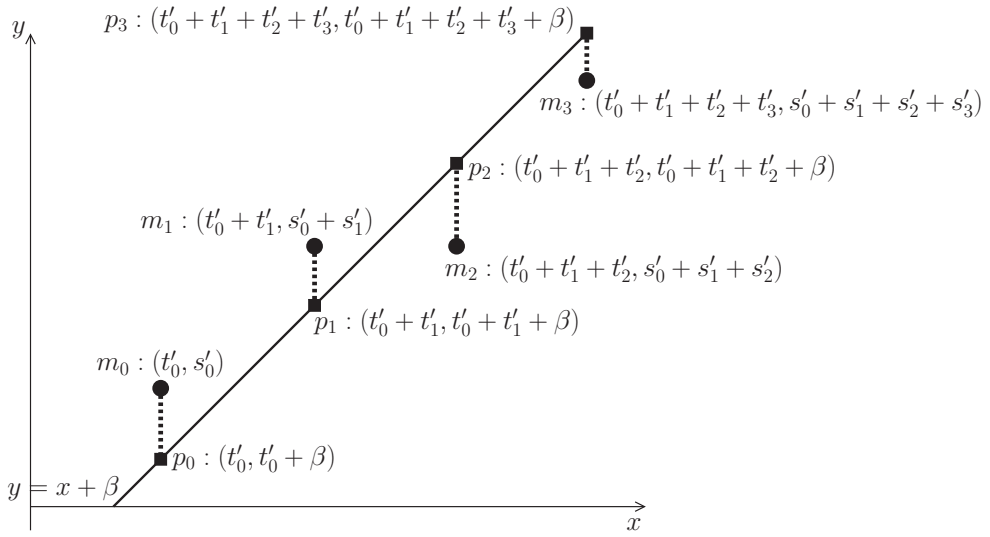


Figure IV.5: An example which depicts how the Shift amounts are estimated. The points $\{m_0, ..., m_{N'-1}\}$ that correspond to the match assignment are marked in solid black dots and the match assignment corresponding points $\{p_0, ..., p_{N'-1}\}$ at $y = x + \beta$ are drawn in solid squares. The Shift amount d_i is measured by subtracting the y coordinate value of p_i from the y coordinate value of m_i.

Then, the change amount δ_i for the *changing operations* can be estimated by solving the following linear equations.

$$\begin{bmatrix} 1 & -1 & 0 & \dots & 0 & 0 \\ 0 & 1 & -1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & -1 \\ -1 & 0 & 0 & \dots & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta_0 \\ \delta_1 \\ \vdots \\ \delta_{N'-1} \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ d_{N'-2} \\ d_{N'-1} \end{bmatrix}.$$

Let $\delta_0 = \alpha$, then

$$\begin{cases} \begin{bmatrix} -1 & 0 & \dots & 0 & 0 \\ 1 & -1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & -1 \end{bmatrix} \begin{bmatrix} \delta_1 \\ \vdots \\ \delta_{N'-1} \end{bmatrix} = \begin{bmatrix} d_0 - \alpha \\ d_1 \\ \vdots \\ d_{N'-2} \end{bmatrix}. \\ \delta_{N'-1} = d_{N'-1} + \alpha = \alpha - (\sum_{i=0}^{N'-2} d_i) \end{cases}$$

Thus,

$$\begin{bmatrix} \delta_0 \\ \delta_1 \\ \delta_2 \\ \vdots \\ \delta_{N'-2} \\ \delta_{N'-1} \end{bmatrix} = \begin{bmatrix} \alpha \\ \alpha - d_0 \\ \alpha - d_0 - d_1 \\ \vdots \\ \alpha - (\sum_{i=0}^{N'-3} d_i) \\ \alpha - (\sum_{i=0}^{N'-2} d_i) \end{bmatrix}.$$

In other words,

$$\delta_0 = \alpha, \text{ and } \delta_j = \alpha - (\sum_{i=0}^{j-1} d_i) \text{ for } j = 1, ..., N' - 1. \tag{IV.8}$$

Since the Deforming distance is defined as the sum of change amount δ_i for all the needed *change operations*. That is the Deforming distance is equal to $\sum_{i=0}^{N'-1} |\delta_i|$. Thus, the Deforming distance can be determined by finding α, such that $\sum_{i=0}^{N'-1} |\delta_i|$ is minimized.

Given a value α, a set of values can be partitioned into two subsets A and B, where elements in A is larger than α and elements in B is smaller than α. Let n_A and n_B are the number of elements in A and B respectively, and $\mathcal{S}$ is the sum of difference between α and elements in A and B. By increasing or decreasing α by σ, the change amount of $\mathcal{S}$, $\Delta\mathcal{S}$, is equal to $|\sigma| \times |n_A - n_B|$. Since $\Delta\mathcal{S} \geq 0$, $\mathcal{S}$ is minimal when $n_A = n_B$. Therefore, α can be determined by assigning its value equal to the Median of $\{0, \delta_0, ..., \sum_{i=0}^{N'-1} \delta_i, \sum_{i=0}^{N'-2} \delta_i\}$. By substituting α to Equation IV.8, the Deforming distance under the constraint of a given match assignment can be estimated.

Global Deforming Distance

To generate all possible match assignments of two given number sequences, the algorithms to solve the *string-to-string correction problem* are considered.

Let $A = \{a_1, a_2, ..., a_n\}$ and $B = \{b_1, b_2, ..., b_m\}$ be two strings of characters and let three edit operations: 1) insertion - insert a character into a

string, 2) deletion - delete a character from a string, and 3) replacement - replace one character by a different character, be used to correct the string A to string B.

Let $A(i)$ and $B(i)$ be two sub-strings which contain the first i characters of A and B, respectively. Since the minimal editing cost from $A(i+1)$ to $B(i+1)$ can be estimated based on the combined cost correcting $A(i)$ to $B(i)$, $A(i+1)$ to $B(i)$, and $A(i)$ to $B(i+1)$. String-to-string correction problem is generally solved by dynamic programming[24]. As shown in Figure IV.7, a diagram is created according to strings A and B. $C(i,j)$ which represents the minimal cost from $A(i)$ to $B(j)$, can be formulated as follows:

$$
\begin{aligned}
C(i,0) &= i, \text{ for } i = 0, ..., n; \\
C(0,j) &= j, \text{ for } j = 0, ..., m; \\
C(i,j) &= min \left\{ \begin{array}{l} C(i-1,j)+1, \\ C(i,j-1)+1, \\ C(i-1,j-1) + \left\{ \begin{array}{ll} 1 & , a_i \neq b_j \\ 0 & , a_i = b_j \end{array} \right. \end{array} \right\}, \text{ for } i = 1, .., n \text{ and } j = 1, ..., m.
\end{aligned}
$$

If $C(i,j)$ is estimated based on $C(i-1,j)$, that means a *deletion* is applied. If $C(i,j)$ is estimated based on $C(i,j-1)$, that means a *insertion* is applied. If $C(i,j)$ is estimated based to $C(i-1,j-1)$, that means a *replacement* is applied, or no operations is performed.

In Section 2.1, three deforming operations - insertion, deletion, and changing are defined to convert a number sequence to another. When considering an *insertion* of deforming operation to be an *insertion* of string-editing operations, a *deletion* of deforming operation to be a *deletion* of string-editing operations, and a *changing* of deforming operation to be a *replacement* of string-editing operations, a match assignment can be represented as a path in a dynamic programming diagram for string-to-string correction problem. Figure IV.8 shows an example of the graphical representation of a match

assignment between two number sequence S and T. In this example, there are one *insertion* next to s_0, and two *deletion* at s_2 and s_{m-1}, respectively.

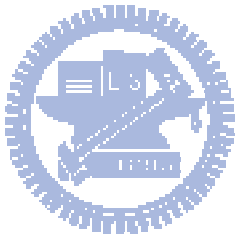
By generating all the possible correction path for the corresponding string-to-string correction problem, possible match assignments of two given number sequences are available. However, dynamic programming is not capable of estimating the deforming distance of two number sequences, due to the deforming distance of two number sequences may not relate to the deforming distance of two sub-sequences of the two original number sequences. For example, given two number sequence $S = \{1, 6, 1\}$ and $T = \{2, 4, 2\}$, the deforming distance is 1. Although, $S' = \{1, 6\}$ and $T' = \{2, 4\}$ are sub-sequences of S and T respectively, the deforming distance between S' and T' is 4/3, which is larger than the deforming distance between S and T. That is, the deforming operations that are used to convert S to T may not include the deforming operations to convert S' to T'.

Let $S = \{s_1, ..., s_n\}$ and $T = \{t_1, ..., t_m\}$ be two number sequences, and $S_i = \{s_i, ..., s_n, s_1, ..., s_{i-1}\}$ be the number sequence created by rotating $i - 1$ elements in S. For each S_i, a Deforming distance between S_i and T is measured using the methods proposed in Section 2.2. The minimum among all the n deforming distances are selected as the global Deforming distance.

Figure IV.9 shows the Deforming distance of all the possible match assignments which convert a number sequence with 3 elements to a number sequence with 2 elements. The graphical representation of each match assignment is shown as a solid line in each block. The brightness of blocks or numerical values under the block represent the minimal magnitude among the local Deforming distance under the constraint of the corresponding match assignment.

As the example shown in Figure IV.9, the global **Deforming distance**

can be found by searching every possible match assignments if the number of normal vectors in a polygon descriptor is small. If the number of normal vectors in a polygon descriptor becomes larger, the global **Deforming distance** can be approximated by using Evolutionary Computation[17].



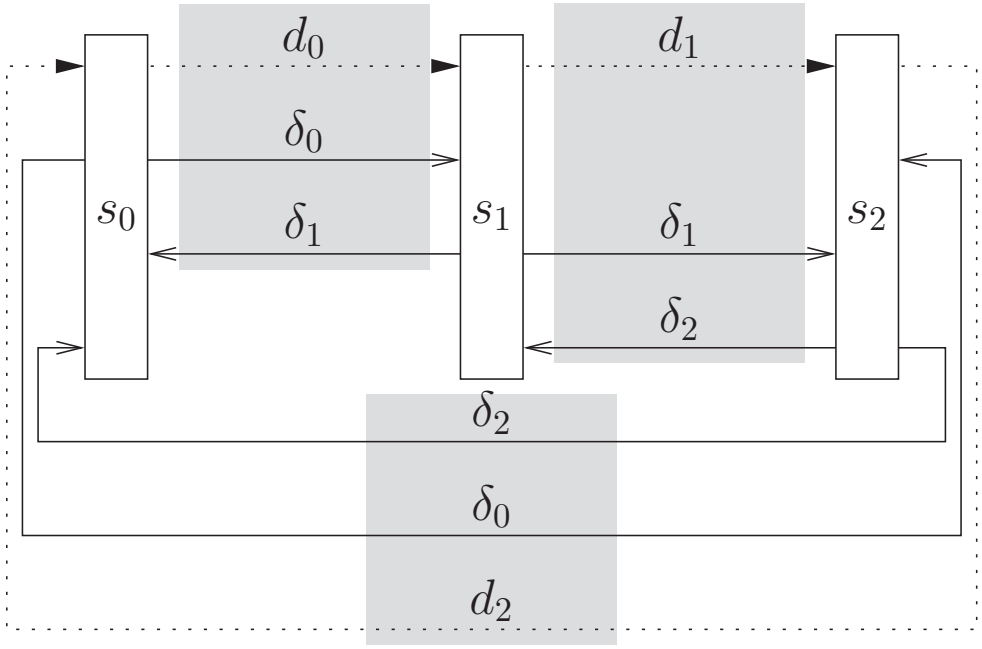


Figure IV.6: An exemplar sequence S' is used to show the relation between shift amounts d_i and the change amounts δ_i for the *chang operation*. The sequence S' contains three elements s_0, s_1, and s_2. Three shift amounts d_0, d_1, and d_2 are moved from s_0, s_1, and s_2 respectively. The movement of change amounts δ_0, δ_1, and δ_2 indicate how to use change operations to achieve the proper shift amounts among s_0, s_1, and s_2. The dotted lines indicate how the partial amount of values d_i are transferred between elements. The solid lines indicate how the *changing operations* distributed partial amount of values δ_i to the adjacent elements. The light gray blocks show the related transfer between adjacent elements in number sequence. Since the amount of values transferred between elements should be the same for d_i and δ_i, thus $d_i = \delta_i - \delta_{i+1}$.

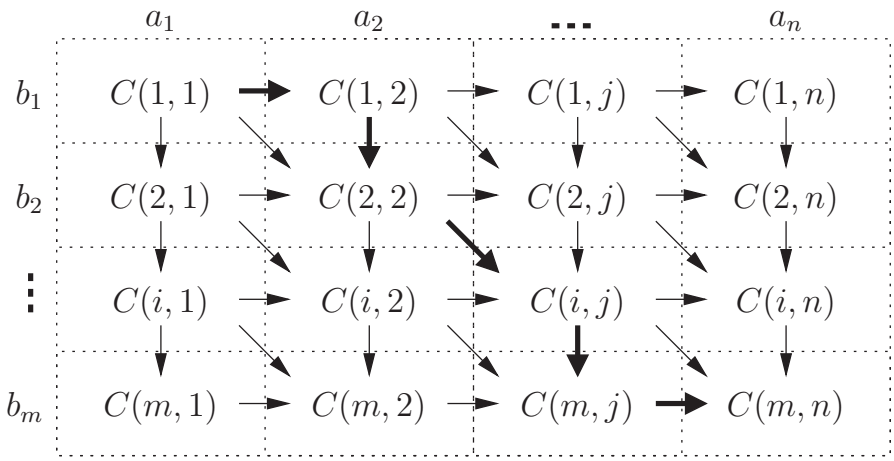


Figure IV.7: Solving the string correction problem by dynamic programming. Let a_i be the i-th character of string A and b_j is the j-th character of string B, and $C(i,j)$ represents the editing cost of to correct the first i characters in A to the first j characters in B. Each arrow represents a dynamic programming operation. The operation starts from the left-top corner and terminates at the right-bottom corner. A minimal cost correction path from the left-top corner to right-bottom corner is searched by the dynamic programming method.

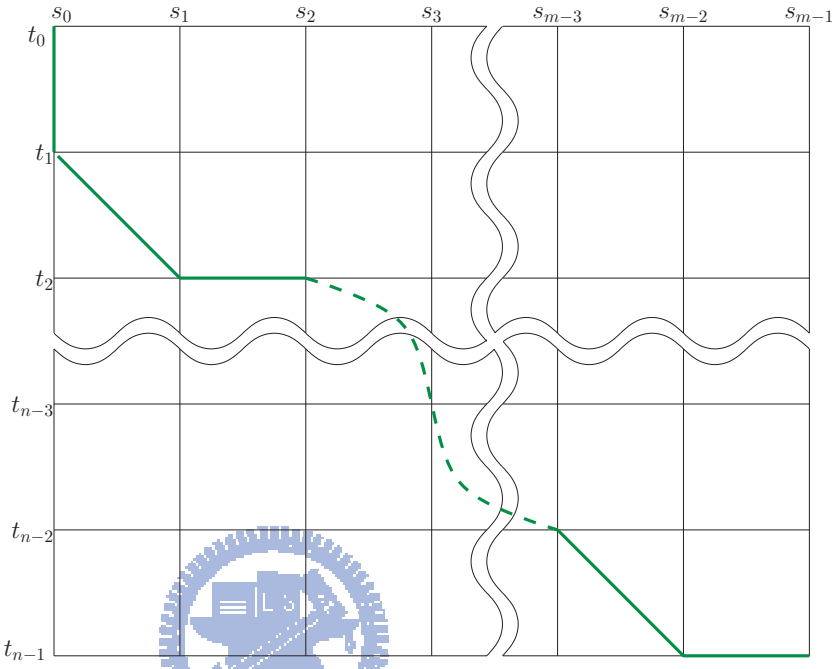


Figure IV.8: An example of the graphical representation of a match assignment between $\{s_0, ..., s_{m-1}\}$ and $\{t_0, ..., t_{n-1}\}$. The graphical representation is similar to a correction-path on the dynamic programming diagram for string-to-string correction problem. In this example, there are one *insertion* next to s_0 and two *deletions* at s_2 and s_{m-1}.

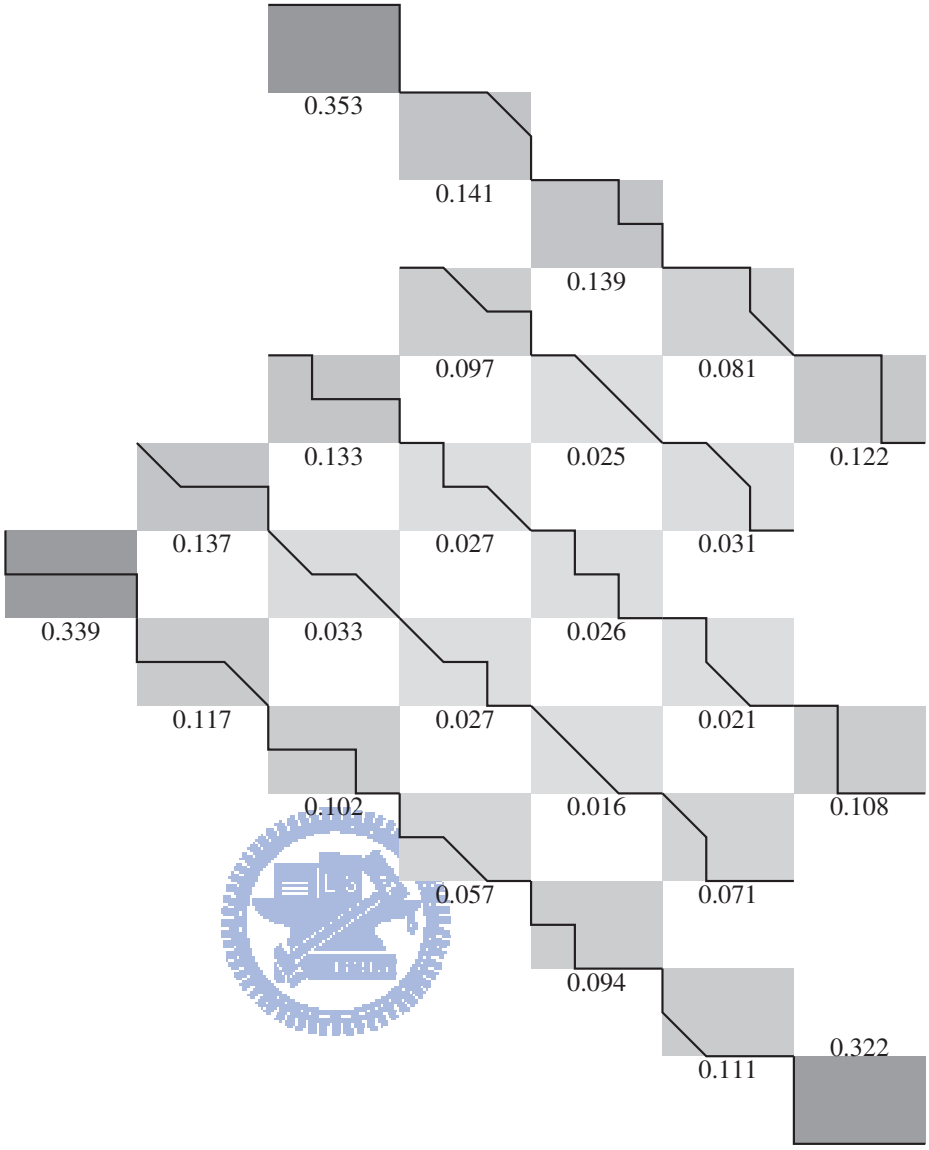
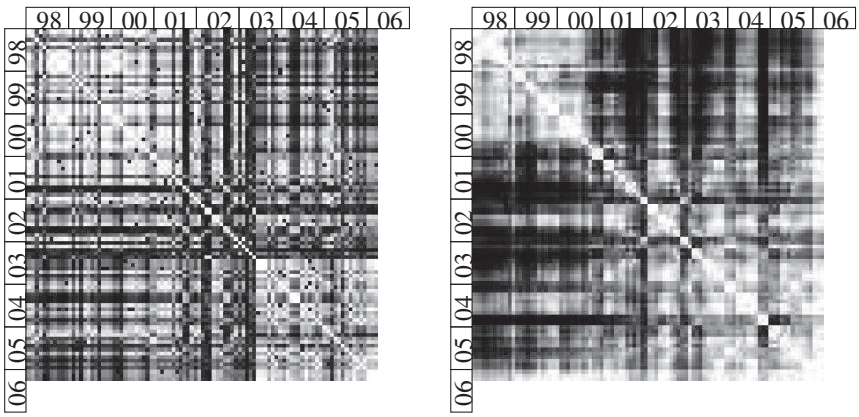


Figure IV.9: The Deforming distance of possible match assignments to convert a sequence containing 3 elements to a sequence containing 2 elements. A block represents the match assignment corresponding a correction path of string-to-string correction problem. The solid line in each block is the graphical representation (see. Figure IV.8) of a match assignment. And the darkness of blocks or the value under the block represent the minimal magnitude among the Deforming distance of the match assignments.

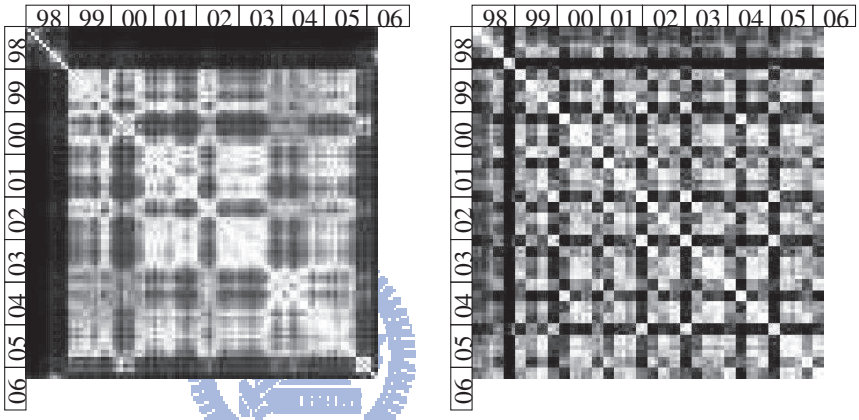
2.3 Evaluation and Experimental Results

According to the news paper and some popular stock investment magazines, Taiwan stock market experienced a clear turning point at the fourth quarter of 2002. In this section, we will use the information as ground truth to evaluate the proposed methods and three other methods, which are: (1) *Hausdorff distance* based, (2) *histogram* based, and (3) *mixture Gaussian* model based methods. The price-EPS data during the period between 1998 and 2002 are used as source data. Four similarity measurement results, which are estimated by using the proposed methods and three other methods, are shown in Figure IV.10. In the following, *PD+MDP* represents the proposed *Polygon descriptor* and *Deforming distance*. *GMM+KLD* represents the method that use *Gaussian Mixture Model* to describe the distribution of data points and *K-L distance* to measure the distance between two Gaussian Mixture Models. *Hausdorff* represents the method that measures the similarity based on the *Hausdorff distance*. And, *Hist+Corr* represents the method that describes the distribution of data points by *histogram* and uses the *correlation coefficient* to represent the similarity between two data sets.

As shown in Figure IV.10, the *Hausdorff-distance* and *histogram* based methods show that 1999 is the end of the first stage, then the market went into another stage immediately. The result pattern of *Gaussian mixture* based method shows that the market finished its first stage after the middle of 2000 (i.e. 30th months after 1998) and started the second stage at the beginning of 2002. The pattern associated with proposed method, *PD+MDP*, shows that the market finished its first stage at July 2001 (i.e 43th months after 1998) and started the second stage at the first quarter of 2003. Apparently, the end of the first stage and the start of the second stage reported by the proposed method is quiet match with the ground truth.



(a) PD+MDP (b) GMM+KLD

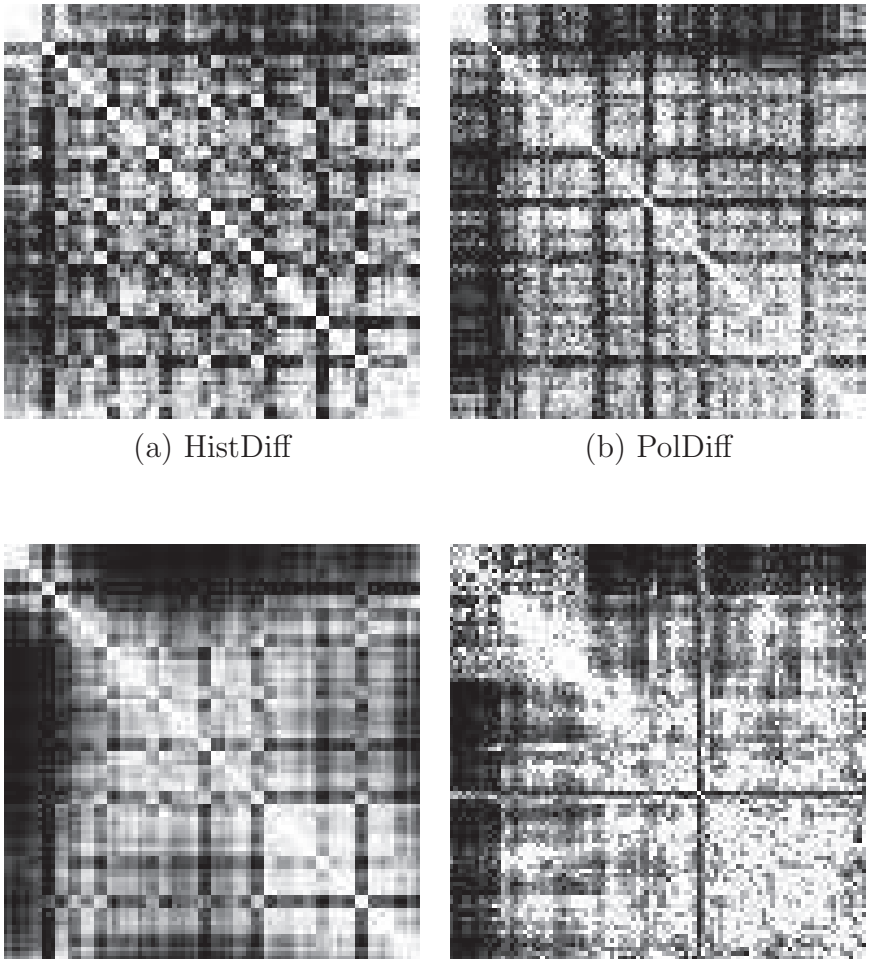


(c) Hausdorff (d) Hist+Corr

Figure IV.10: The resulted similarity pattern of the proposed method and three common used methods. For each similarity pattern, there are 96 columns and 96 rows corresponding to 96 months price-EPS data from January 1998 to December 2005. (a) pattern of *PD+MDP* shows the similarity results created by using Polygon descriptor and Minimal Deforming distance. (b) pattern of *GMM+KLD* shows the similarity results created by using Gaussian Mixture model and K-L distance. (c) pattern of *Hausdorff* shows the similarity results created by using Hausdorff distance. And, (d) pattern of *Hist+Corr* shows the similarity results created by using histogram and correlation measurement.

Among the 4 methods under evaluation, only the proposed method can show the turning points of stock market behavior in a precise manner. We believe this measurement accuracy comes from the inherent invariant naturals, i.e., translation, scale and rotation invariant in the Polygon descriptor. These invariant operations make the proposed method to concentrate on core stock market information and to ignore surface and/or human made inferences while performing similarity measurement. However, the other three methods use the coverage area techniques for data modeling, which usually are sensitive to some surface disturbances and/or human inference on stock price information.

Figure IV.11 shows more similarity matrices by using the other methods introduced in Section 1. **HistDiff** is the similarity matrix created using histogram to represent the data distribution and measuring the similarity between histograms by residual. **PolDiff** is created by using polar histogram instead. And **PolCorr** is created by polar histogram and correlation coefficient. **CSQ** is created by using Chi square error to measure the difference between mixture Gaussian models. Obviously, these results by using these methods are no good.



(a) HistDiff (b) PolDiff

(c) PolCorr (d) CSQ

Figure IV.11: Four resulted similarity matrices created by - a) histogram+residual, b) polar histogram+residual, c) polar histogram+coefficient correlation, and d) Gaussian mixture model+chi square error.

3 Minimal-Cost Normal Vectors Match

As shown as the example given in Figure IV.12, $\mathcal{X} = \{X_1, X_2, X_3, X_4\}$ and $\mathcal{Y} = \{Y_1, Y_2, Y_3, Y_4\}$ are sets of normal vectors for two polygon descriptors. For each normal vector, the angle magnitudes between it and the other normal vectors are measured to represent the normal vector. That's, $X_i = \{x_{ij} \| j \neq i\}$, where x_{ij} is the angle magnitude between X_i and X_j. The similarity between $\mathcal{X}$ and $\mathcal{Y}$ can be defined as the minimal cost to create a multiple-to-multiple onto match from normal vectors in $\mathcal{X}$ to normal vectors in $\mathcal{Y}$. As shown as the example given in Figure IV.14, a multiple-to-multiple onto match from $\mathcal{X}$ to $\mathcal{Y}$ is searched according to the similarities between elements in $\mathcal{X}$ and $\mathcal{Y}$ to represent the similarity between two polygon descriptors. The match in this example represents that Y_1 and Y_2 are split from X_1, X_2 and X_3 are merged to Y_3, and X_4 becomes Y_4.

As shown as the example given in Figure IV.13, the similarity between two normal vectors X_i and Y_j is measured by finding a one-to-one onto[19] match with maximal sum of similarity $Sim(x_{ik}, y_{lj})$.

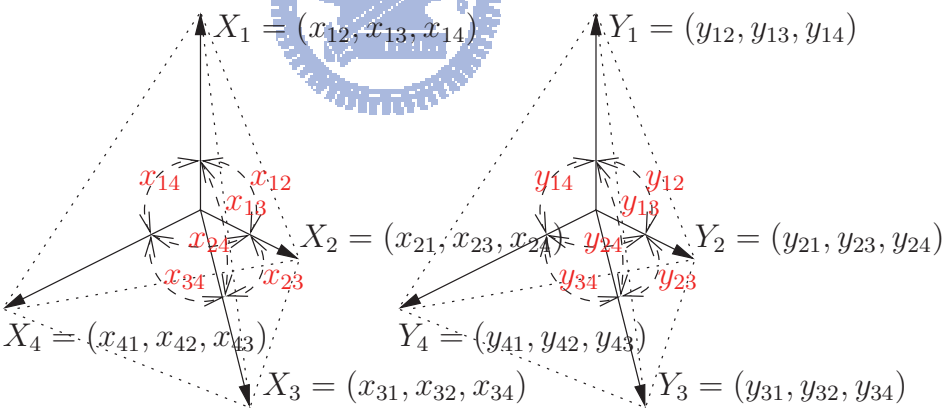


Figure IV.12: For each normal vector, the amplitude of angles between adjacent normal vectors are measured to represent the normal vector.

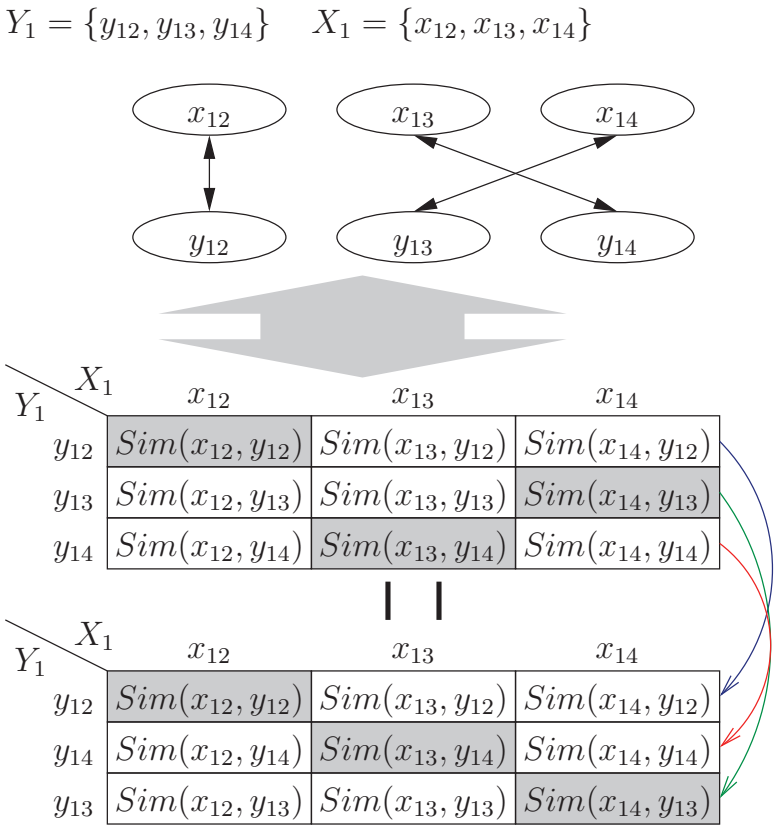
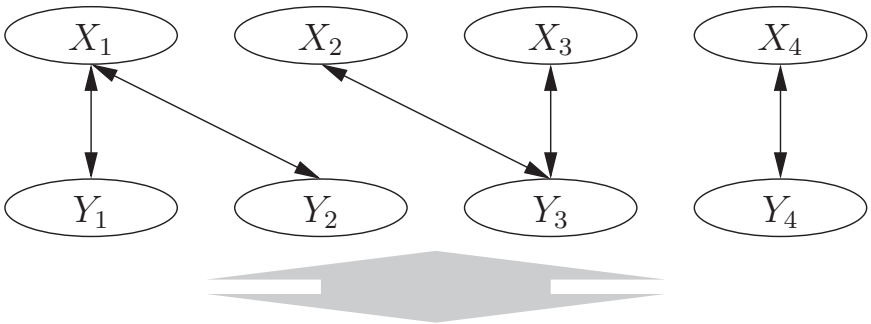


Figure IV.13: For two normal vectors X_1 and Y_1, a similarity matrix for any two angles in two corresponding angle values is computed first. Then the similarity of two normal vectors are measured by finding a minimal cost one-to-one onto match

3.1 One to One Onto Matching

For each normal vector, the angle magnitudes between it and the other normal vectors are measured to represent the normal vector. To measure the similarity between two normal vectors, a one-to-one and onto match[19] between elements in two set of angle magnitudes can be performed by the following proposed method.

Giving two set of angle magnitudes $X_i = \{x_{i1}, ..., x_{iN}\}$ and $Y_j = \{y_{j1}, ..., Y_{jM}\}$.



$Sim(X_1, Y_1)$	$Sim(X_2, Y_1)$	$Sim(X_3, Y_1)$	$Sim(X_4, Y_1)$
$Sim(X_1, Y_2)$	$Sim(X_2, Y_2)$	$Sim(X_3, Y_2)$	$Sim(X_4, Y_2)$
$Sim(X_1, Y_3)$	$Sim(X_2, Y_3)$	$Sim(X_3, Y_3)$	$Sim(X_4, Y_3)$
$Sim(X_1, Y_4)$	$Sim(X_2, Y_4)$	$Sim(X_3, Y_4)$	$Sim(X_4, Y_4)$

Figure IV.14: The overall similarity between two Polygon Descriptors are measured by finding a minimal cost multiple-to-multiple onto match.

An $N \times N$ matrix M is built as follows,

$$M(X_i, Y_j) = \begin{bmatrix} m_{11} & \cdots & m_{1N} \\ \vdots & \ddots & \vdots \\ m_{N1} & \cdots & m_{NN} \end{bmatrix}, \text{where } m_{kl} = S(x_{ik}, y_{jl}),$$

where $S(x_{ik}, y_{jl})$ is the similarity between two angle magnitudes x_{ik} and y_{jl}.

Since an element m_{kl} represents the similarity value when x_{ik} and y_{jl} are considered to be matched with each other. Finding an one to one and onto match between X_i and Y_j is equivalent to selecting N elements from each column of M, and no two elements are from the same row. To have the largest overall similarity of a one-to-one and onto match between X_i and Y_j, the N elements should be selected such that the sum of selected elements is maximized. In order to regularize the computing procedure, row exchanges are performed to have the N selected elements aligned on the diagonal position. Thus, the similarity value between two set of angle magnitudes can be computed as the sum of diagonal elements.

Figure IV.15 shows the flow chart of the proposed method that maximize the sum of diagonal elements by properly exchanging rows. Giving a similarity matrix M, the method is performed as follows. Find an element m_{ij} with the largest value. Swap the i-th row and j-th row such that the element m_{ij} is at (j, j). Then mark all elements in j-th row and j-th column. Repeatedly find elements with maximal value from un-marked elements and exchange the maximal element to diagonal position until all elements are marked.

Although the procedure stated above are efficient, its results may not be good enough. Therefore, a decision step is applied on each rows and columns to find if any possible row exchange can further maximize the sum of diagonal elements in the similarity matrix M. The decision step is described as follows,

IF $m_{ii} + m_{jj} > m_{ij} + m_{ji}$ THEN exchange the i-th and j-th rows.

This decision step is performed repeatedly until no further row exchange can be performed to increase the sum of diagonal elements. In other words, the near maximum similarity value $S(X_i, Y_j)$ between normal vectors $\mathcal{X}_\rangle$ and $\mathcal{Y}_|$ is achieved.

An example of the one-to-one onto match method is shown in Figure IV.16. At first, since $m_{11} = 178$ is the largest value in the 5×4 matrix, the elements in first row and first column are marked. Since the value $m_{23} = 169$, the largest value among the unmarked elements, is located in the third row, the second row and the third row are exchanged. Then the elements in second row and second column are marked too. Similar process are repeated until all columns are marked. According to the decision step, the second and the fourth rows are exchanged since $80 + 143 > 60 + 162$.

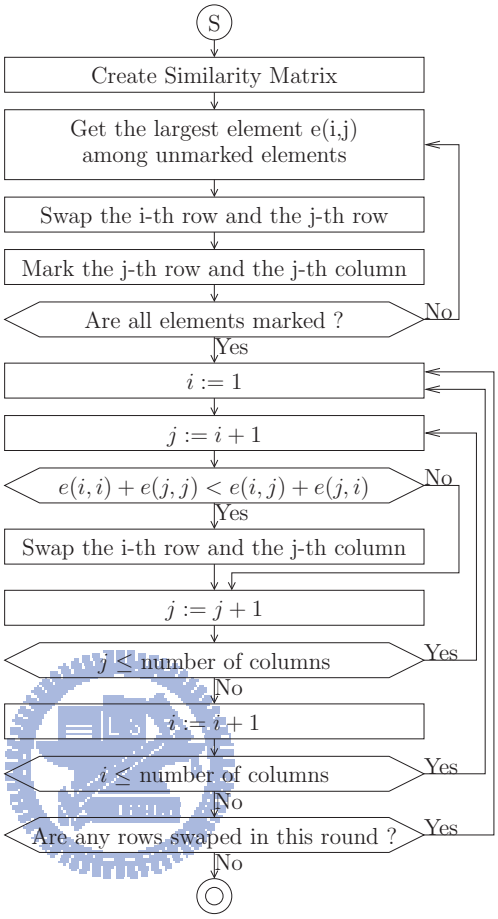


Figure IV.15: The flow chart of the proposed matching method. At the beginning, a similarity matrix is constructed. The order of rows are then initialized by greedily exchanging the largest element to diagonal position. Then further decision steps are applied on the diagonal elements to maximize the result.

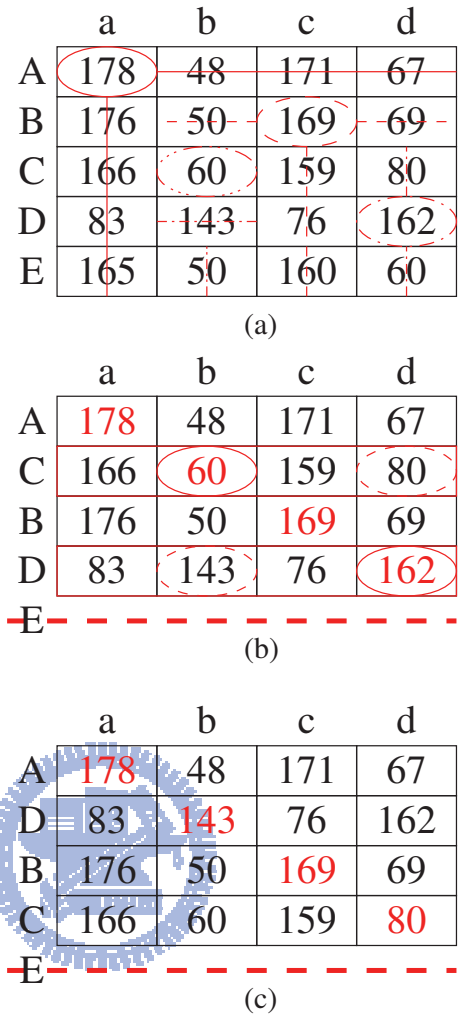


Figure IV.16: An example of the proposed matching method. The similarity matrix constructed from the pair-wise similarity values is shown in (a). By greedily picking the smallest elements from selectable elements of similarity matrix and exchanging them to the diagonal position, the order of rows in matrix is initialized as shown as (b). Then the decision steps are applied to find more row exchanges to maximize the sum of diagonal elements. The final results are shown in (c).

3.2 Multiple to Multiple Onto Matching

By referencing the bipartile graph matching algorithm stated in [25], a greedy method is proposed to find a best multiple-to-multiple onto match between two set of normal vectors.

According to the similarity between normal vectors (see Section 3.1), a dis-similarity matrix $\mathcal{M}$ is created as follows,

$$\mathcal{M}(\mathcal{X},\mathcal{Y}) = \begin{bmatrix} m_{11} & \cdots & m_{1N} \\ \vdots & \ddots & \vdots \\ m_{N1} & \cdots & m_{NN} \end{bmatrix}, \text{where } m_{ij} = 1 - S(X_i, Y_j).$$

Then a set of elements in $\mathcal{M}$ is selected to let every row an column of dis-similarity matrix $\mathcal{M}$ contains at least one selected elements and the sum of selected elements is minimized.

Figure IV.17 shows the flow chart of proposed multiple-to-multiple onto match method. At the beginning, the dis-similarity matrix $\mathcal{M}$ is created by using the similarity values between normal vectors. First, an element $e(i,j)$ having largest value among all unmarked elements are selected. Then all selected elements are checked and modified to decrease the total value of selected elements in constraint of that all matrix rows and columns containing selected elements still contain at least one selected element.

Figure IV.18 shows an example to introduce the proposed multiple-to-multiple onto match method. At the beginning, the largest value 0.02 is selected. Then 0.08, 0.28, and 0.29 is selected, too. When 0.35 is tested, although the third row and first column already contains 0.08 and 0.28. 0.35 is still selected to replace 0.08 and 0.28, because $0.35 > 0.08 + 0.28$ and there are other elements in the second column and the fourth row.

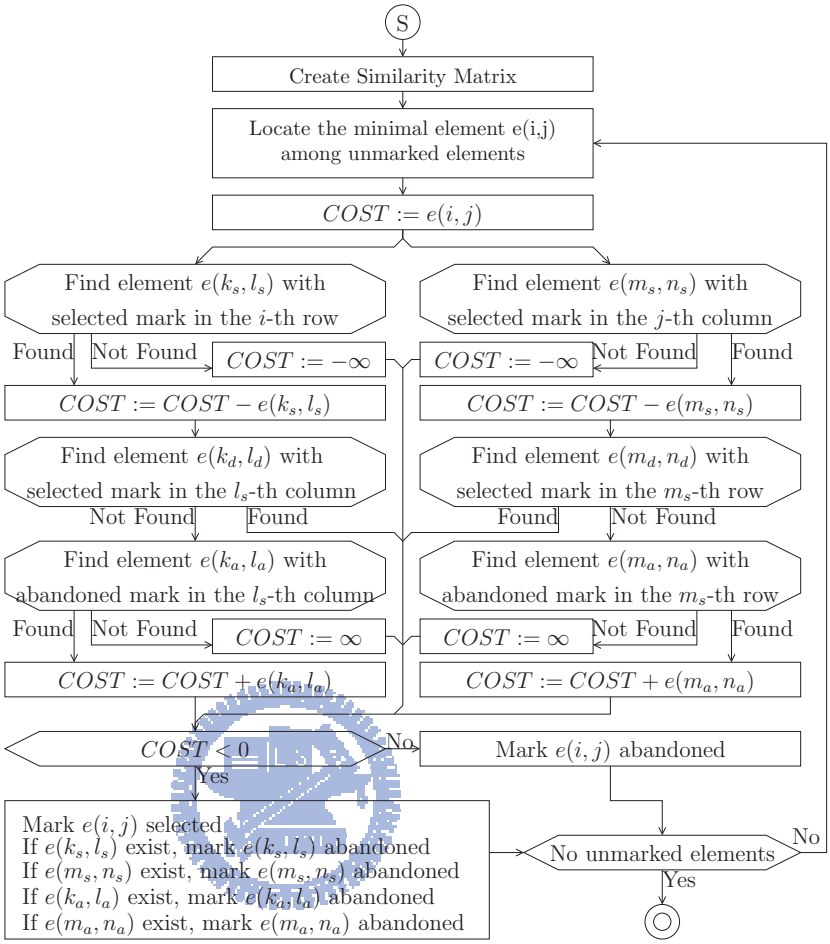


Figure IV.17: The flow chart of the proposed multiple-to-multiple onto match method. At first, a dis-similarity matrix $\mathcal{M}$ is created by using the similarity between normal vectors. Then the elements in dis-similarity matrix $\mathcal{M}$ are checked from large to small to find matches having minimal sum of values in constraint of that all matrix rows and columns contain at least one selected element.

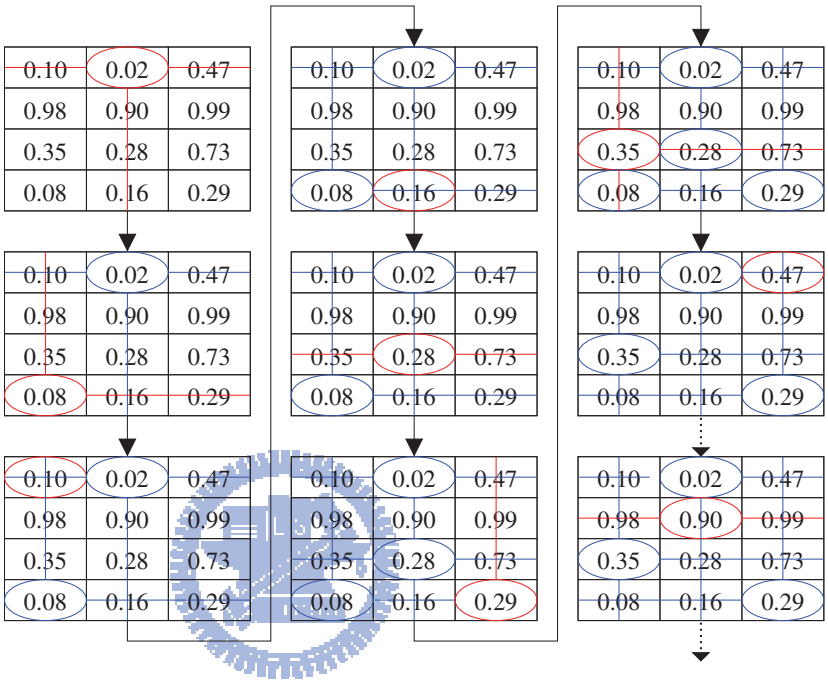


Figure IV.18: An example of multiple-to-multiple onto match method.

3.3 Evaluation and Experimental Results

In order to evaluate the performance of the proposed one-to-one and onto match algorithm, synthesis testing data sets are generated by a random number generator. To evaluate different size of similarity matrices, 10 groups of test data sets with row dimension from 2 to 11, are generated. For each group, random seeds from 0 to 5999 are used to generate test data.

The testing results are listed in Table IV.1. **OST**(Optimizing Sorting Test) is performed by the proposed method. **TAT**(Testing All Test) is performed by an exhaustive search method which validates all the possible solutions and the **TAT** results are considered as the ground truth. The performance is measured by the total spending time in milliseconds to process all the test data. Apparently, the proposed method is quite efficient. Besides, based on the 60000 (10 group of 6000 random seeds) testing data, the proposed method generates the same results as the ground truth.

In addition, the proposed multiple-to-multiple onto match method is also evaluated by 60000 random number sets. The spending time of the proposed method (**GMT**) and the exhaustive searching method (**ESM**) are listed and compared in Table IV.2. Obviously, the proposed method is efficient when comparing to the exhaustive searching method. Besides, the error rate is only 0.00165(99/60000).

By applying the proposed methods on stock price and EPS (earn per share) data collected during the period from 1986 January to 2006 March, the similarity matrix is created. Monthly stock price and earn per share(EPS) data are collected during the period from 1986 January to 2006 March. Then, collected stock price and EPS data are used to create data distributions of stock price and EPS for each month. After modeling the shape of monthly data distribution by Polygon descriptor, the proposed **Minimal-Cost Nor-**

Table IV.1: The performance comparison of the proposed one-to-one match algorithm (OST) and the exhaustive search method (TAT). To evaluate different size of similarity matrices, 10 groups of test data sets with row dimension from 2 to 11, are generated. The data listed in table are the averaged computing time in milliseconds spent by the matching process. TAT/OST is the ratio of the spending time by these two methods.

Matrix Dimension	Computing Time (in milliseconds)		
	TAT	OST	TAT/OST
2	206	175	1.2
3	295	214	1.4
4	855	216	4.0
5	5440	228	23.9
6	40853	232	176.1
7	349117	255	1369.1
8	3582596	301	11902.3
9	39197125	327	119868.9
10	469318748	362	1296460.6
11	6219635476	408	15244204.6

mal Vector Match method is used to measure the similarities between data distributions of every two months. A similarity matrix is created by using the measured similarities between monthly data. The brightness of a pixel at (i, j) in similarity matrix represents the similarity between i-th and j-th months since 1986 January. As the results shown, the similarity matrix is still capable to reflect the behavior change of market.

Table IV.2: The performance comparison between the proposed multiple-to-multiple onto match method (**GMT**) and exhaustive searching method (**ESM**). By using 60000 random number sets, matrix in different dimension(2×2,3×3,4×4, and 5×5) are created and tested using the two methods. When the matrix dimension is very small, **ESM** performs better. However, when the matrix dimension gets larger, the performance of the proposed method (**GMT**) is better than exhaustive searching method (**ESM**) obviously.

Matrix Dimension	Computing Time (in milliseconds)		
	ESM	GMT	ESM/GMT
2	347	667	0.5202
3	9634	1480	6.5095
4	1993646	3035	656.8850
5	1568824936	5065	309738.3882

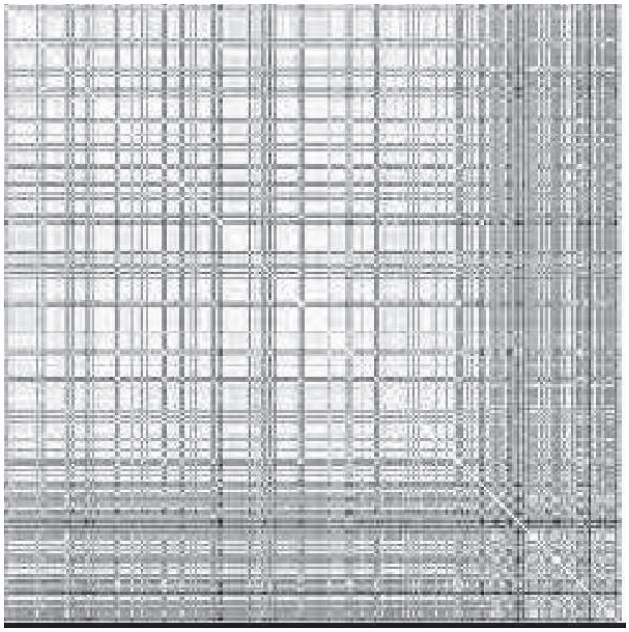


Figure IV.19: The similarity matrix created using the stock price and EPS (earn per share) data collected during the period from 1986 January to 2006 March. The brightness of pixel (i, j) represents the similarity between the i-th and j-th month since 1986 January.