CHART RECOGNITION AND INTERPRETATION IN DOCUMENT IMAGES

ZHOU YANPING

NATIONAL UNIVERSITY OF SINGAPORE

2003

CHART RECOGNITION AND INTERPRETATION IN DOCUMENT IMAGES

ZHOU YANPING

(Ph.D Candidate, NUS)

A DISSERTATION SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

NATIONAL UNIVERSITY OF SINGAPORE

2003

Name: Zhou Yanping
Degree: Doctor of Philosophy
Dept: Department of Computer Science
Dissertation Title: Chart Recognition and Interpretation in Document Images

Abstract

In graphics recognition, chart recognition and interpretation is a procedure to change scientific chart images into computer readable form. In this dissertation, we have investigated four problem domains in it. First, we propose a hierarchical statistical-model-based framework for chart recognition system. Second, we propose an improved projection-based plot area detection method to detect plot areas and a Hough-based axis detection algorithm to detect axes. Third, we propose a new approach for chart classification and segmentation based on statistical modeling. A novel chart classification approach based on Hidden Markov Models is proposed. A new approach for chart segmentation using optimal path finding is also proposed. Fourth, we propose a novel structure called zoned directional X-Y tree to hierarchically represent the text primitives in charts. An algorithm of generating the zoned directional X-Y tree is presented. Both results from chart segmentation and text primitive analysis are correlated for chart interpretation.

Keywords: Graphics Recognition
Chart Recognition and Interpretation
Hough Transform
Statistical Modeling
Hidden Markov Model
Zoned Directional X-Y Tree

Acknowledgements

I would like to express my heartfelt gratitude and appreciation to my supervisor Professor Tan Chew Lim for the advice and guidance he has provided throughout my PhD work. I would also like to thank him for his great patience and encouragement. He has been most approachable and helpful throughout the period.

I would like to thank Professors Leow Wee Kheng and Sung Kah Kay for their advice and guidance during my graduate studies. I am grateful to Professor Blostein for the instrumental discussion on chart recognition when I attended the 1st conference of Diagram. I would like to thank members of thesis committees.

I am indebted to many of my colleagues and friends who have given me their support and encouragement during my research work, especially to Long Huizhong, Zhang Qinjun, Tang Menting, Xu Yi, Michael Cheng, Zhang Yu, Zhijian, Fusheng, Wang Bin, etc.

Finally, this dissertation could not been possible without the support of my loving family: my parents Zhou Baigen and Wu Facong, my husband Tom and my lovely son Edward. I am forever grateful for their love, patience, and measureless support.

This dissertation is dedicated to my father Zhou Baigen.

谨以此书献给我的父亲周柏根！

Table of Contents

List of Figures

List of Tables

Summary

Chart recognition and interpretation is a procedure to change scientific chart images into computer readable form such as tabular data. Unfortunately there is little work reported on it due to the difficulties and challenges in four main issues: the great diversity of chart types, the flexibilities in the structural arrangement, the difficulty in describing the syntax and semantics of complex charts and the difficulty in dealing with degraded, distorted or noisy input.

In this dissertation, we have investigated four problem domains in chart recognition: chart recognition system, chart graphic symbol extraction, chart classification and segmentation, text primitive analysis and chart interpretation.

Chart recognition system: We propose a hierarchical statistical-model-based framework for scientific chart recognition system. First, the knowledge of chart generation software is explored and notation conventions of a scientific chart from both generation and recognition point of views are defined. Second, investigation in psychological aspect and human visual perception on charts deduces three arguments that are the backbone of the proposed framework. Our testing data is constructed with more than 500 chart images from technical journals that are scanned by 300 dpi.

Chart graphic symbol extraction: Chart graphics symbol recognition of current work includes plot area detection and axis detection. We propose an improved

projection-based plot area detection method to detect plot areas. For axis detection, we propose a Hough-based axis detection algorithm that combines geometric analysis of 2-D and 3-D axes.

Chart classification and segmentation: We propose a new approach for chart classification and segmentation based on statistical modeling. Four chart models including separated bar model, contiguous bar model, single-line-series line model and multiple-line-series line model are constructed and trained using a segmental *K*-means algorithm to model the semantics of chart stage area. Charts are classified by choosing the chart model with the largest posteriori probability. The best state path for that model is also obtained by applying Viterbi algorithm. Two kinds of classifications, dimension classification and type classification, are addressed. We also propose a new approach for chart segmentation using optimal path finding. Two chart segmentation problems are addressed, including detecting the number of data series and bar pattern segmentation.

Text primitive analysis and chart interpretation: We propose a zoned directional X-Y tree structure to hierarchically represent the text primitives in charts. An algorithm of generating the zoned directional X-Y tree is presented. The algorithm includes three procedures: directional transformation of the bounding boxes, recursive X-Y cut by the bounding boxes and linking the bounding boxes with the X-Y tree. A scheme combining X-Y tree searching and traversing with structural analysis is proposed to label the text primitives in a chart. Three kinds of axes tick labels are extracted: vertical axes tick labels, horizontal axes tick labels and directional axes tick labels. The extraction of the axes titles and the figure titles is also presented. Finally, both the result from chart segmentation and text primitive analysis are correlated for chart interpretation.

Chapter 1

Introduction

1.1 Motivation

In our society today, paper is still an important medium for exchanging information in literary, scientific or commercial fields. Most of the paper-based documents are in the raster file style. Thus by changing the paper-based documents into a computer readable electronic format, it can broaden the scope of our information source. The growing need for information sharing among different work and research communities and the development of new technologies for digital information diffusion have increased the demand for tools for automatically converting paper-based information into computer readable information.

As far back as 1985, it was stated that about one trillion statistical graphs were printed each year [114]. Many more of such graphs are expected with the proliferation of printed paper documents today. Most of statistical graphs appearing in scientific papers are scientific charts or diagrams. Like forms or tables which convey information from structurally arranged data, scientific charts are also a very powerful representation tool in the scientific research area because people understand symbolic graphs better and faster

than the corresponding text [115]. The processing procedure to change scientific chart images into computer readable form is *scientific chart recognition*. The ensuing processing procedures like understanding the meaning of the scientific charts or changing recognized electronic charts into other computer readable forms such as tabular data form are in the field of *scientific chart interpretation*. There is little research work and practical products reported on recognizing and interpreting scientific chart images in comparing with those on the table or form recognition. In the next section, we discuss the challenges and difficulties in recognizing and interpreting scientific chart images that lie in the following main four aspects.

1.2 Challenges

The Great Diversity of Chart Types

Many text-processing software packages have built-in features or tools for generating charts and graphs, such as Microsoft Excel and Word, Harvard Graphics, Corel Chart, etc. 2-D or 3-D graphical objects such as lines, circles, rectangles, cones, cylinders, pyramids and spheres are used in these scientific chart generation tools as one of the customized features. Figure 1.1 shows some chart types used in the Microsoft Excel Chart tools. Charts can be classified into color charts or monotonic charts. Combinational charts in which different chart graphical objects are used to present complex data also appear frequently in the data presentation. Different patterns and textures can also be used for filling the graphical objects like bars and pies to denote different categories in the scientific charts. For instance, there are 48 textured patterns in the chart generation tools of Microsoft Excel chart tools as shown in figure 1.2. The color variations for the

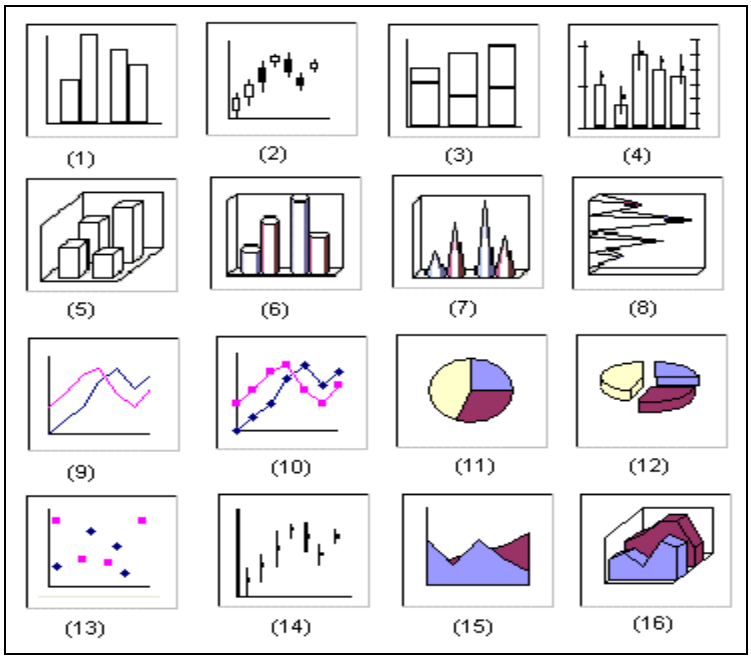


Figure 1.1: Some chart types in the Microsoft Excel Chart tool: (1): clustered column. (2): open-high-low-close. (3): stacked column. (4): volume-high-low-close. (5): 3-D column. (6): column with a cylindrical shape. (7): column with a conical shape. (8): column with a pyramid shape. (9): line. (10): line with markers displayed on each data point. (11): pie. (12): pie with a 3-D visual effect. (13): scatter. (14): high-low-close. (15): area. (16): area with a 3-D visual effect.

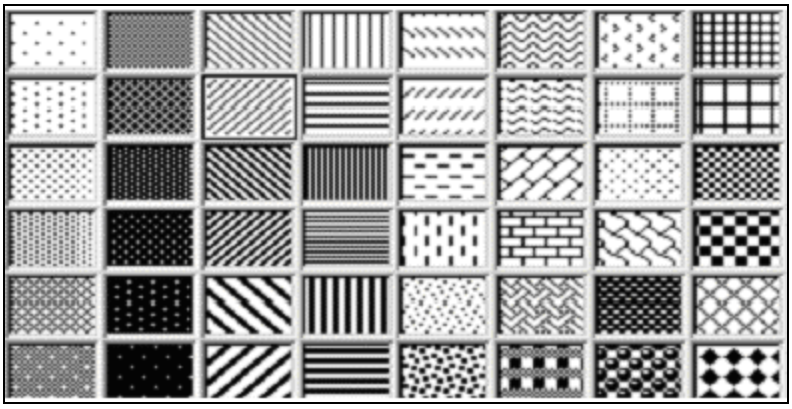


Figure 1.2: Filling patterns in the chart generation tool of Microsoft Excel. There are 48 texture patterns that can be applied on the surfaced graphical objects such as bars, columns, pies and areas.

foreground and the background inside each pattern can give birth to a large number of colorful patterns. Consider applying 64 colors on the textured patterns. There will be 193,536 colorful textured patterns generated. Therefore for a simple bar chart with only one data series, different colorful textured patterns in the bars lead to a total of 193,536 different bar charts, not to mention bar charts with several data series.

The Flexibilities in the Structural Arrangement

Even in the same chart type, charts may look very different from each other due to the positional translation or rotation of graphical or text objects. For example, most chart generation tools offer users with various customization functions, such as putting the title at an arbitrary position of the chart, etc.

The Difficulty in Describing the Syntax and Semantics of Complex Charts

While most of the two-dimensional charts have simple syntactic and semantic meaning like bar charts and line charts, the meaning for most of the three-dimensional charts is always difficult to describe for further chart recognition or interpretation.

The Difficulty in Dealing with Degraded, Distorted or Noisy Input

Poor image quality introduced by an inappropriate acquisition of an image such as bad illumination, noise introduced by an external device or vibrations in the acquisition device, image degrading caused by previous processing steps, increases the difficulty of a recognition procedure. Typical degradations appearing in the document image are: gaps due to the lack of ink which causes the discontinuity of lines, extra large noise caused by ink blobs, or image warping at the left or right side caused by uneven scanning, etc.

Thus to generate a generic type-independent chart recognition system is a highly challenging problem. The difficulties led us to the research objectives given in the next section.

1.3 Research Objectives

The task of meeting the challenges set out in the preceding subsection is indeed daunting and is not very much researched so far in the document image analysis community. It is impossible to address the entire problem within the time frame of the present dissertation.

With a practical scope in mind, this dissertation aims to investigate four problem domains in chart recognition by investigating the recognition and interpretation of two major kinds of charts: bar charts and line charts. Furthermore, it consists of four main objectives:

1. Chart recognition system: Propose a sound scientific chart recognition framework and theoretical analysis for the foundation of the proposed chart recognition framework.
2. Chart graphic symbol extraction: Investigate two intermediate-level graphical processing procedures: plot area detection and axes detection.
3. Chart classification and segmentation: Investigate two kinds of chart classification: dimension classification and type classification. Dimension classification is to classify a chart into a 2-D chart or a 3-D chart. Type classification is to classify a 2-D chart into one of the four chart categories: the single-line-series chart, the multiple-line-series chart, the separated bar chart and

the contiguous bar chart. Chart segmentation involves two issues: detect the number of data series and bar pattern segmentation.

4. Text primitive analysis and chart interpretation: The problem of labeling the structural texts in a chart is also explored. Text primitive analyses involving extraction of the axes tick labels, the axes titles and the figure titles are proposed in our work. The segmented axis tick labels are essential for interpreting a chart and transferring chart data into a tabular output by correlating with the value points from chart segmentation.

1.4 Contributions and Dissertation Outline

We aim to make contributions from four problem domains that we will investigate in this dissertation: chart recognition system, chart graphic symbol extraction, chart classification and segmentation, text primitive analysis and chart interpretation.

In the problem domain of chart recognition system, the contributions will be as follows:

1. We will propose some notation definitions of a scientific chart from a recognition point of view. Notational conventions from both generation point of view and recognition point of view facilitate the whole chart recognition procedure.
2. We will give theoretical contributions in constructing a chart recognition system by investigating the mechanism of human visual perception on chart recognition. We will examine the arguments that form the principles and backbone of our chart recognition problems.
3. We will propose a hierarchical statistical-model-based chart recognition framework which focuses on the intermediate level of vision.

4. We will collect a large set of test data. The procedure of setting up the testing data for our system is not difficult but tedious. In future work, the test data set will be made publicly available for future studies.

In the problem domain of chart graphics symbol recognition, the contributions will be as follows:

5. We will propose an improved projection-based approach for plot area detection.
6. We will present a method of axes detection with Hough feature clustering and geometric analysis in our work to detect 2-D and 3-D axes.

In the problem domain of chart classification and segmentation, the contributions will be as follows:

7. We will propose a new framework for chart classification and segmentation based on statistical modeling.
8. We will propose a model-based chart classification approach. This includes feature extraction with feature point segmentation and analysis, construction and train of HMM-based chart models, type classification by chart model matching.
9. We will propose a new approach for chart segmentation by optimal path clustering and finding.

In the problem domain of text primitive analysis and chart interpretation, the contributions will be as follows:

10. We will propose a zoned directional X-Y tree structure to hierarchically represent the text in graphical documents. The proposed zoned directional X-Y tree is a generalized version of the classical X-Y tree which considers only orientations in the vertical and the horizontal directions.

11. We will propose a method of directional transforming the bounding boxes in the image space to the r-space.
12. We will propose a recursive X-Y cut segmentation algorithm using original and transformed bounding boxes to generate the zoned directional X-Y tree for text primitives.
13. We will present an approach of combining X-Y tree searching and traversing with structural analysis to label the text primitives in a chart. Detailed procedures to extract axes tick labels and titles will be illustrated.
14. We will present a method of correlating value points with axis tick labels in order to interpret chart data into a tabular format for bar charts and line charts.

The above targeted contributions will be addressed in the dissertation which is outlined below:

A survey of graphics recognition and related works will be conducted in chapter 2. Chart recognition system will be addressed in the chapter 3. In chapter 4, intermediate-level chart graphical processing such as plot area detection and axes detection are proposed. Chart classification and segmentation using statistical modeling are presented in the chapter 5. In chapter 6, the problems of text primitives analysis and chart interpretation are addressed. We conclude the dissertation and point out the further directions of our work in chapter 7.

Chapter 2

Related Works

Scientific chart recognition is a branch of the application of *graphics recognition* which in term is a sub-area of *document image analysis* (DIA). Document image analysis is "the study of converting documents from paper form to an electronic form that captures the information content of the document" [10] and " the practice of recovering the symbolic structure of digital images scanned from paper or produced by computer" [82].

The wide ranging research interests and topics due to the great variety of the document contents have led to the emergence of the field of document image analysis. These active studies and practices are classified into two main categories in terms of the document contents: one is the mostly-text DIA such as optical character recognition [55, 96, 105], handwritten character recognition [48, 70, 80] and document layout analysis [63, 103], etc. The other category is the mostly-graphics DIA, namely, graphics recognition. Within the last two decades, we have seen conferences and workshops organized for the sole purpose of document image analysis research. These include the *international conference on document analysis and recognition* (ICDAR), the *international workshop on document analysis systems* (DAS), the *international workshop on graphics recognition* (GREC), the *SPIE conference on document recognition and*

retrieval, etc. A new journal, namely, *international journal of document analysis and recognition* (IJDAR) also came into being following the growing interest in the field. Comprehensive surveys and research studies on the document image analysis can be found in [3, 11, 12, 41, 82, 107].

2.1 Graphics Recognition

Although text is no doubt the major source of document data, a large number of graphs, photographs, pictures, and diagrams are also accessible in our daily lives. Just like the old adage that "a picture is worth a thousand words", information in pictorial representation is much more complex and unwieldy than that in textual representation. Graphics are complex and difficult to interpret for machines, while machines can recognize characters quite easily.

We focus on graphs and diagrams which are concise and abstract pictorial representations of information. Maps, scientific charts, engineering drawings, and sketches are all examples of graphs and diagrams. For example, people use scientific charts such as line charts and bar charts to intuitively convey a clear analysis of commercial data and research data. In architecture and engineering design, the technique of computer aided design (CAD) is extensively used to produce a large number of engineering drawings, electrical circuit diagrams, flow charts and process diagrams to facilitate the communication among human designers, producers and engineers. The goal of *graphics recognition* is to convert information from its paper-based graphical representation into computer interpretable data.

The area of graphics recognition has undergone extensive investigation for several decades. But there are still many tough problems unsolved in this field [24, 58, 82, 110].

We first review some graphics recognition systems in the next section.

2.1.1 Graphics Recognition Systems

Works in many specific application domains of graphic recognition have been reported, such as circuit diagram recognition, geographical map recognition, engineering drawing recognition, fingerprint classification etc.

- **Engineering Drawings Recognition**

Yu *et al.* [123] presented a system to recognize a large class of engineering drawings which include flowcharts, logic and electrical circuits, and chemical plant diagrams. First, domain-independent rules are used to segment symbols from connection lines in the drawing image that has been thinned, vectorized, and preprocessed. Second, a drawing understanding subsystem works with a set of domain-specific matchers to classify symbols and correct errors. The final output of the system is a net list of identified symbol types and interconnections

Unlike routine engineering drawing recognition systems which use a two-pass vectorization work flow that introduces more propagation error, Song *et al.* [104] proposed an efficient one-phase object-oriented vectorization model that recognizes each class of graphic objects from raw images. Each graphic object is recognized directly in its entirety at the pixel level. The raster image is progressively simplified by erasing

recognized graphic objects to eliminate their interference with subsequent recognition. The experiments and evaluation results show significant improvement in speed and recognition rate.

Other works and survey papers on engineering drawings can be found in [46, 53, 74, 75, 109], etc.

- **Circuit/Structure Diagram Recognition**

In [33], Fahn *et al.* designed a topological–based electronic circuit diagram recognition. His objective was to extract circuit symbols and characters. Line segments were detected and approximated using a segment-tracking algorithm and a piecewise linear approximation algorithm. A topological search was performed to separate them into symbols and characters. Context-based depth-first-search approach was to classify all the symbols.

Casey *et al.* proposed a prototype for encoding chemical structure diagrams in [14]. The structure diagram first was separated using size and spacing characteristics. Line drawings were discriminated and the meaning of bonds was obtained after vectorization. Atomic symbols were classified using chemical drawing conventions and optical character recognition.

Other works on circuit diagram can be found in [31, 85], etc.

- **Maps Recognition**

In [44], Hartog *et al.* proposed a knowledge-based framework to interpret and segment maps from top to down. First the image was segmented globally. Then a top-down segmentation combining the contextual reasoning framework was performed to

detect the inconsistency. The interpretation of maps is based on a priori knowledge of maps.

Oshitani and Watanabe [86] discussed a pipeline-like recognition strategy based on information propagation to address the boundary and synchronization problems caused by parallel processing on the partitioned map images.

Other recent works and survey papers on maps recognition can be found in [98, 99, 119], etc.

- **Music Scripts Recognition**

Fahmy and Blostein [32] applied a graph-rewriting technique to recognize printed musical scripts. The recognition starts from vectorized music primitives. Layout constraints are handled by the graph-rewriting paradigm for discrete relaxation where neighborhood-construction is interleaved with constraint-application. Approximately 180 graph-rewriting rules are used to express notational constraints and semantic-interpretation rules for simple music notation.

Blostein and Haken [9] also investigated ways to use diagram generators to improve diagram recognizers. The Lime music-notation editor, a generator for music notation authored by Blostein and Haken, has been used to correct over 80% of the note-duration errors made by MIDIScan, a commercial recognizer for music notation.

Other recent works and survey papers on music scripts recognition can be found in [8, 62, 106], etc.

- **Mathematics Recognition**

Zanibbi and Blostein [124] proposed an efficient typeset and handwritten mathematical notation recognition system. They used a tree transform technique to direct the high level recognition of mathematical notation. Three levels of processing procedures are proposed. In the first Layout Pass level, a Baseline Structure Tree (BST) is constructed from a list of symbols with bounding boxes. Next, a Lexed BST is generated from the initial BST by grouping the low level input symbols into complex tokens and then translated into LATEX in the Lexical Pass level. Finally, additional domain-specific processing is applied to produce output for symbolic algebra systems in the Expression Analysis Pass level.

Other recent works and survey papers on mathematics recognition can be found in [17, 18, 25, 26, 56, 79, 108], etc.

- **Tables/Forms Recognition**

Yu and Jain [122] proposed a generic form recognition system using *block adjacency graph* (BAG) to the extraction of form frames and preprinted data. The strokes of the characters that overlap the frame are reconstructed after removal of the line. Templates are constructed from empty forms and correlated with filled-in forms. With the form template, the system can recognize both handwritten and machine-typed filled-in forms.

Fan *et al.* [34] presented a clustering-based approach to recognize form document. Characters are first extract from the form using feature points clustering method on the feature points from thinning. Next, a clustering process is applied to the corner points of the remaining structured line patterns. Form document is then represented as a weighted graph according to the clustering result for further graph matching.

Other recent works and survey papers on tables and forms recognition can be found in [2, 4, 50, 73, 76, 96, 102, 125], etc.

- **Other Diagrams Recognition**

Kasturi *et al.* [57] developed a system to interpret line drawings. Many graphics techniques are exploited such as connected components, collinear component grouping, thinning, boundary tracking, loop analysis, to identify outline and solid polygonal objects and their spatial relationships, such as circular arc segments, hatched areas, dashed lines, connectors between objects and text strings, etc.

Francesconi *et al.* [36] developed an adaptive recursive neural network model to recognize logos. First, logos with exterior or interior contour features are represented in contour-tree format which contains the structural information of logos and continuous attributes of contour nodes. The contour−tree features are then used to train and recognize logos through recursive neural networks.

Other works on other diagrams recognition can be found in [28, 38, 65, 84, 93], etc.

2.1.2 Methodology of Graphics Recognition

The sequence of processing steps for graphics recognition is similar to that of common document image processing. Many review papers have proposed the basic structures of graphics recognition system from different points of view [7, 29, 82]. The whole graphics recognition system can be roughly divided into two levels of processing: graphics symbol objects recognition and graphics symbol arrangement analysis [7].

- **Graphics Symbol Recognition**

While more detailed surveys on graphics symbol recognition can be found in [24, 40, 69, 72], we give a brief survey below:

The symbol objects recognition is the lower level processing in the whole graphics recognition processing system. Usually it consists of four steps: preprocessing which includes noise reduction and de-skew operations, vectorization, symbols segmentation, symbols classification and recognition.

There are three different approaches to symbol objects recognition: template matching recognition, deformable template matching recognition and learning-based recognition.

Template matching recognition usually comprises of segmenting symbols, vectorization and generating a description file and finally model matching to get the best matched symbols [1, 33, 67]. In [1], Ah-Soon proposed a constraint network for symbol detection in architectural drawings. First the symbols are described as a set of constraint rules of segments and arcs. A description language is used to describe the rules. Symbols are detected by propagating the segments and arcs in the network in order to search for the matched models. In [67], Lee proposed a model based method to recognize the electrical circuit symbols. An attribute graph encoding structural and statistical features is used in model matching.

In deformable template matching recognition, the template or the model is variable to some degree [13, 77, 116]. Messmer and Bunke [77] presented a model-based method combining pattern recognition and machine learning techniques to recognize and learn the graphics symbols in engineering drawings. First, vectorized line drawings and graphics symbols are represented in the attributed relational graph format and stored in the database. The matching models are organized in a network format. In the graphics

symbols processing, a sub-graph isomorphism recognition method is performed to give the optimal match. Finally, the detected symbols are grouped into the database using heuristic rules.

Learning-based recognition is usually segmentation-free. In this approach, the features extracted for training and testing are selected before segmentation. Therefore, the inaccuracy caused by the segmentation processing can be avoided [15, 23, 36]. In [23] Cheng *et al.* developed a hierarchical neural network based segmentation- free symbol recognition for electrical drawings. The invariant geometric features are selected for the neural classifiers for further processing. In [15], Cesarini *et al.* developed a neural network based system to locate and recognize the low level graphics items. Graphics items are first located by morphological operations and connected component analysis. Then an auto-associator-based neural network is used to classify the located items. Symbol recognition using Hidden Markov Models (HMM) can also be classified into learning-based category since each HMM needs training before being used to recognize new symbols. The applications based on Hidden Markov Models are reviewed in the section 2.2.2.

- **Graphics Symbol Arrangement Analysis**

Graphics symbol arrangement analysis constitutes the high level graphics recognition processing. The structural and logical relationships between the symbols are explored in this level processing. Besides heavy domain-specific knowledge needs to be applied to this procedure for a robust system. In [7], Blostein gave a detailed analysis on the general frameworks of symbol arrangement analysis and classified different approaches into five categories of framework: blackboard framework, schema-based framework,

syntactic-based framework, computational vision framework and graph-rewriting framework. These five categories are briefly explained below.

In the first category, blackboard framework can be also called hypothesis testing framework, in which knowledge sources are incorporated into different hypothesis levels with confidence values. A rule-based inference engine for interpreting rules is usually used to further search for the best symbol arrangement scheme. For instance in [59], Kato and Inokuchi developed a four-level blackboard-based system to recognize circuit diagrams. The four levels are input diagrams, symbol hypotheses, diagram hypotheses and recognition results.

In the second category, schema-based framework usually defines a schema class to represent the diagram prototype. Two kinds of information, spatial relationship and object composition, are incorporated into the schema class. Also a strategy grammar like rule-based inference engine is used to direct the searching or parsing of the schemas. The Mapsee systems developed by Mulder *et al.* [81] are one example of schema-based diagram recognition systems. Mapsee uses a *scene constraint graph* to represent the knowledge of sketch maps. Constraints are applied and propagated when parsing the scene constraint graph and directing the interpretation of the sketch maps.

In the third category, syntactic-based framework uses grammar to represent diagram domain knowledge, which is composed of a start symbol and a set of rewriting rules. By using top-down or bottom-up parsing, the production rules were combined with the spatial constraints and the grammar to partition the symbols into related groups. For example, Chou [25] used syntactic stochastic grammars to recognize the noisy mathematical expression images.

In the fourth category, computational vision framework goes with a high-level computation related with hierarchical views of vision [112].

In the fifth category, graph-rewriting framework first uses graph representation to construct the knowledge base. The nodes of the graph encode the information of symbol objects in the graphics image, such as the attributes of the objects. The edges of the graph are used to build the relationships among the objects. There are also attributes associated with the edge. A graph grammar that is composed of an initial graph and graph transition rule sets is used to direct the parsing and searching scheme. For example, in [32], Fahmy and Blostein used graph-rewriting rules to recognize the music notations.

Besides the above five categories, statistic-model-based frameworks such Hidden Markov Model based framework are receiving more and more interests in the field of document image analysis. Works related with this approach are reviewed at section 2.2.2.

2.1.3 Scientific Chart Recognition

Research works for scientific chart or business chart recognition reported are not as many as other diagrams recognition.

In [37], Futrelle and Nikolakis presented a diagram understanding system by constructing graphics constraint grammars for different types of diagrams with syntactic analysis. The work focuses on high level arrangement analysis. The symbol arrangement analysis can be classified into the syntactic-based framework. The basic assumption for his work is that the segmentation is successfully implemented and vectorized primitives are extracted by other vectoriztion tools before applying graphics constraint grammars analysis. The major work he reported is on x, y data graphs and gene diagrams.

In [121], Yokokura and Watanabe proposed a layout-based network which is a schema-based framework to graphically describe the layout relationship information of the bar chart. During the process of symbol object recognition, he used simple vertical and horizontal projection to do segmentation and combine bar chart layout information while extracting graphical and text primitives. In his work, the interleaving of segmentation and classification improves the accuracy of bar chart recognition. But due to the simplicity of the segmentation method, the types of bar charts that can be recognized are constrained by many assumptions. Scanned bar chart images are tested and performance is reported.

Our work in this dissertation is to recognize and interpret scanned chart images. Futrelle and Nikolakis' work assumed that the primitive extraction is completed thus vector representation of graphics primitives is already available for further processing. Thus Yokokura and Watanabe's work is closer to our research.

2.2 Other Related Techniques

In this section, we review some of the computer vision and statistical techniques which we apply in our chart recognition.

2.2.1 Hough Transform

The Hough transform is very closely related to Radon transform. The principle of the Hough transform is to detect geometric shapes by utilizing a voting mechanism to estimate parameters that represent different shape such as line, circle, etc. A brief introduction of Hough transform can be found in appendix A. The Hough transform was

first formulated in 1962 by Hough [47]. Since its presentation, it has undergone intensive investigations that have resulted in several generations and a variety of applications in computer vision and image processing area [51].

The Hough transform is further generalized to detect circle, ellipses or arbitrary shapes [5, 30, 66, 88]. The probabilistic classes of the Hough transforms, such as the probabilistic Hough transform [6] and the random Hough transform [120], were developed to reduce the computation time by minimizing the proportion of points that are used in the voting scheme. Wahl [118] proposed an algorithm by structurally analyzing the cluster patterns in Hough space to interpret 3-D scenes of polyhedral objects. Collinear features are first clustered in the Hough space and construct a structure called *Hough net*. Based on the analysis of the Hough net, hypotheses about the 3-D interpretation are inferred and represented as an attributed graph structure.

There are also many applications of Hough transform in the research area of document image analysis such as handwritten character recognition [22], collinear text grouping [35], feature extraction in graphics recognition [71, 113], etc.

2.2.2 Hidden Markov Model

Hidden Markov Model (HMM) is a probabilistic modeling tool for time series data. There are many tutorials on it [27, 49, 68, 91, 92]. We also give a brief description on it in the appendices of the dissertation.

Hidden Markov Models have been successfully applied in speech recognition [49, 54, 68], part-of-speech tagging [64] and image processing [45]. In the area of document

image analysis, successful applications in handwritten character recognition are also reported [20, 21, 39, 48, 80].

Kopec and Chou proposed the influential *document image decoding* (DID) approach based on statistical Hidden Markov Models in [61]. DID system contain three elements: an image generator, a noisy channel and an image decoder. The model of document recognition process comprises of a message source, an imager and a noise channel. The message source defines the knowledge encoded in a document. The imager is modeled as the mapping scheme from a message source to a noise free image. The channel transforms the noisy free image to an observed image. Given the observed image, the decoder estimates the message source by finding the most probable path. The approach is applied to the problem of decoding scanned telephone yellow pages to extract names and numbers from the listings. Satisfactory results are obtained. The DID approach was also successfully applied to music notation recognition [62]. Unlike DID based on a generative (source) model, Stückelberg and Doermann proposed a descriptive recognition model based on HMM for the task of document understanding [106].

Chapter 3

Chart Recognition System

3.1 Analysis of Scientific Charts

A chart is a symbolic representation of data or information, such as a bar chart, a pie chart or a flow chart, etc. A bar chart is a chart displaying values as vertical or horizontal bars. A pie chart uses a pie and its slices to represent data. The representation element in a line chart is a line. Unlike the abovementioned charts that represent data or values, a flow chart is a diagram that shows the steps in the execution of a program. A scientific chart is a chart using graphics elements such as bars, pies, or lines, etc as the principle presentation elements to present data in a highly meaningful form. Pie charts, dot charts, bar charts are all examples of scientific charts. On the other hand, a flow chart or a structure chart is not in the category of the scientific chart. In our research, our focus is on the scientific charts.

Despite the diversity in chart types, scientific charts show some structural similarity from the perspective point of view. Blostein and Haken [9] suggested using diagram generators to improve diagram recognizers. To understand the features of a chart, we first examine the notational conventions defined by the chart generation tools in order to develop a chart recognition system. To figure out how a chart generation tool creates a

chart is also helpful for developing a chart recognition system. In the following section, we introduce the knowledge of charts from the point view of chart generation.

3.1.1 Knowledge from the Microsoft Excel Chart Tool

Commercial chart creation tools like chart wizard in Microsoft Excel have their detailed notational conventions for charts. Microsoft Excel chart tool is crude but sufficient for recognition as machine will not handle fine details anyway. The chart wizard creates a chart from a two-dimensional data sheet or a table. The dimension with fewer rows or columns is the value dimension, normally along the Y-axis. The dimension with more rows or columns is the category dimension, usually along the X-axis. Chart items include category axis, value axis, data series, data marker, data label, tick marks, tick-mark labels, legend, or gridlines, etc. A real chart may comprise of all of them or part of them. Some charts may also include complementary items such as a secondary value axis. We briefly define the main chart items and areas of a chart as follows. Detailed notational conventions related to them can also be found in the help sources of the Microsoft Excel software [78]. Figure 3.1 also gives an example illustration for these entity definitions.

Chart area: The entire chart and all its elements.

Axis: A line that borders one side of the plot area, providing a frame of reference for measurement or comparison in a chart.

Plot area: The area that is bounded by the axes. The plot area in our work includes the axes.

Data marker: A bar, area, dot, slice, or other symbol in a chart that represents a single data point or value that originates from a worksheet cell. Related data markers in a chart constitute a data series.

Data series: This refers to a group of related data points that are plotted in a chart. Each data series in a chart has a unique color or pattern and is represented in the chart legend. One can plot one or more data series in a chart. Pie charts have only one data series.

Tick marks: Tick marks are small lines of measurement, similar to divisions on a ruler that intersect an axis.

Gridlines: The lines that are added to a chart to make it easier to view and evaluate data. Gridlines extend from the tick marks on an axis across the plot area.

Data label: A label that provides additional text information about a data marker, which represents a single data point or value.

Title: This refers to the descriptive text that is automatically aligned to an axis or centered at the top of a chart. See the chart title, X-axis title and Y-axis title in figure 3.1.

Tick-mark labels (also called tick names): Tick-mark labels identify the categories, values, or series in the chart.

Legend: A box that identifies the patterns or colors that are assigned to the data series or categories in a chart.

The element entities can be classified into two main categories: graphics elements and text elements. Graphics elements include axes, data markers, data series, tick marks, and gridlines. Text elements comprise of various titles and labels.

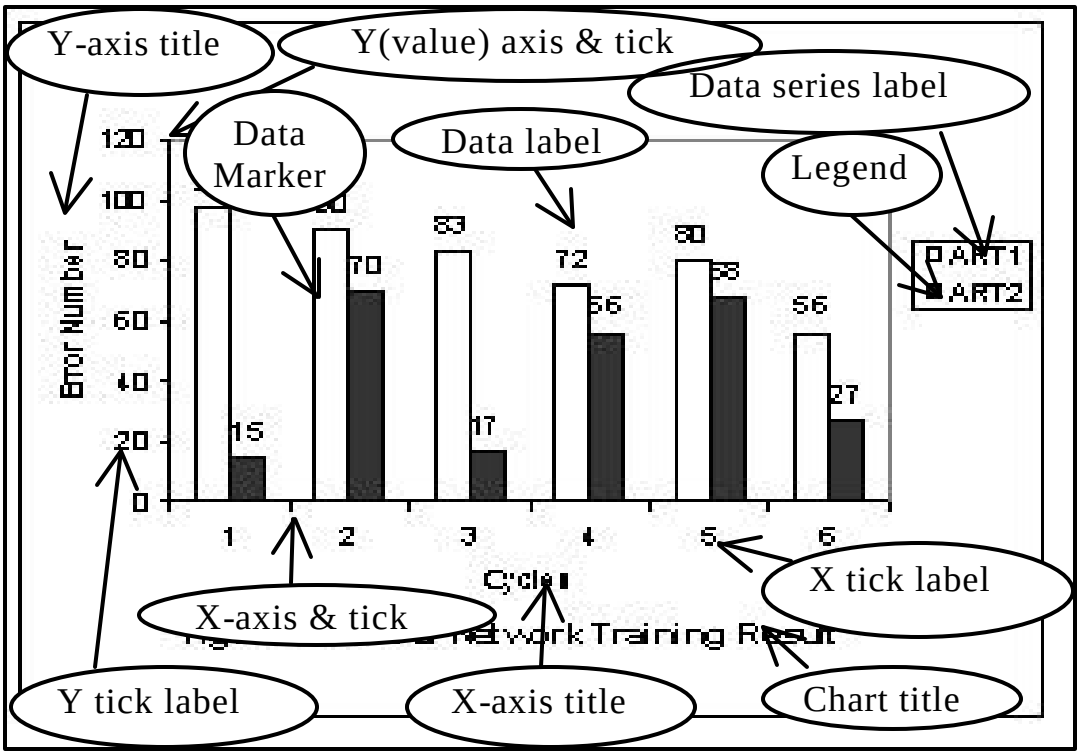


Figure 3.1: Element entities in a chart. Note: the element entities of a chart are indicated in the oval

3.1.2 Definitions

Versatile chart creation tools can create tremendous chart variations by varying the type, format and placement of a chart. On the other hand, the process of converting a raster chart image to a table or other computer readable format is by no means an easy task. While creating a chart from data is a direct process, the reverse process, i.e. from chart to data is a great challenge to us. To design such a conversion system, we have analyzed the structure of bar charts [127] in the light of the chart creation process described in the above section. The same notational conventions in chart creation tools introduced in the above section are adopted in our work. In addition, we also give new definitions or different definitions from the image processing and recognition point of view. We first define two kinds of primitives in our processing: graphics primitives and text primitives.

Graphics primitives: These are the graphics objects that have highly meaningful form. Axes, tick marks, data markers such as bars, lines, are all the graphics primitives.

Text primitives: These are the character objects that are grouped in a highly meaningful form. Titles, labels are examples of the text primitives.

The axes in our work have different names from that of the Excel chart tool. We also define four types of chart areas inside a chart: substrate area, stage area, axes major areas and axes minor areas. The substrate area includes the axes major areas and the axes minor areas.

Axes: For most 2-D charts, data values are plotted along the value axis, which is usually vertical and named Y-axis in our work. Categories in most 2-D charts are plotted along the category axis, which is usually horizontal in 2-D charts and also

named X-axis. In the 3-D charts, the vertical axis is called Z-axis. The axis which is close to the Z-axis is the X-axis. The remaining axis is the Y-axis.

Stage area: This refers to an area where the data markers such as bars or lines are active. The stage area is similar to the plot area defined in Excel. In a 2-D chart, the plot area is the same as the stage area. But in a 3-D chart, the plot area includes more elements than the stage area.

Substrate area: This refers to the part of chart that is composed of the chart frame and its annotations. The annotations include the titles, tick-marker labels, legends, which are not inside the stage area. In other word, the substrate area is the chart area that excludes the stage area.

Axes major areas: The part in the substrate area where the text labels related with axes lie in. For example, the axis tick labels and the title of the axis lie in this area. According to different axes, axes major areas are divided into the X-axes major area, the Y-axes major area, and the Z-axes major area.

Axes minor areas: The part in the substrate area that is parallel to and opposite to the axes major areas. Axes minor areas also can be divided into the X-axes minor area, the Y-axes minor area, and the Z-axes minor area.

We define the following categories of charts according to different classification criterion.

Two-dimensional-axes charts: Charts that have two axes inside it. The angle between them is in the range of 85° and 95°.

Three-dimensional-axes charts: Charts that have three axes inside it. The largest angle between them is larger than 100°.

Single-data-series charts: Only one series of data markers appears in these 2-D charts such as a single line chart.

Multiple-data-series charts: More than one series of data markers appear in these 2-D charts that gives a complex comparison relationship between different series.

According to the above definition, those charts that have data markers with 3-D visual effect but have the largest angle between the axes less than 100° are classified as two-dimensional-axes charts. Thus, a pie chart is either a two-dimensional-axes chart or a three-dimensional-axes chart. In this dissertation, if it is not explicitly explained, the two-dimensional (2-D) charts means two-dimensional-axes charts and three-dimensional (3-D) charts means three-dimensional-axes charts.

There are some terms related to chart interpretation level.

Value points: Points convey the value information of the data marker on a particular position. The value in the horizontal direction of a value point is called its ***horizontal value***. Likewise, the value in the vertical direction of a value point is called its ***vertical value***.

Figure 3.2 is a definition illustration in a two-dimensional-axes chart. Figure 3.3 gives an example of the defined areas in a three-dimensional-axes chart.

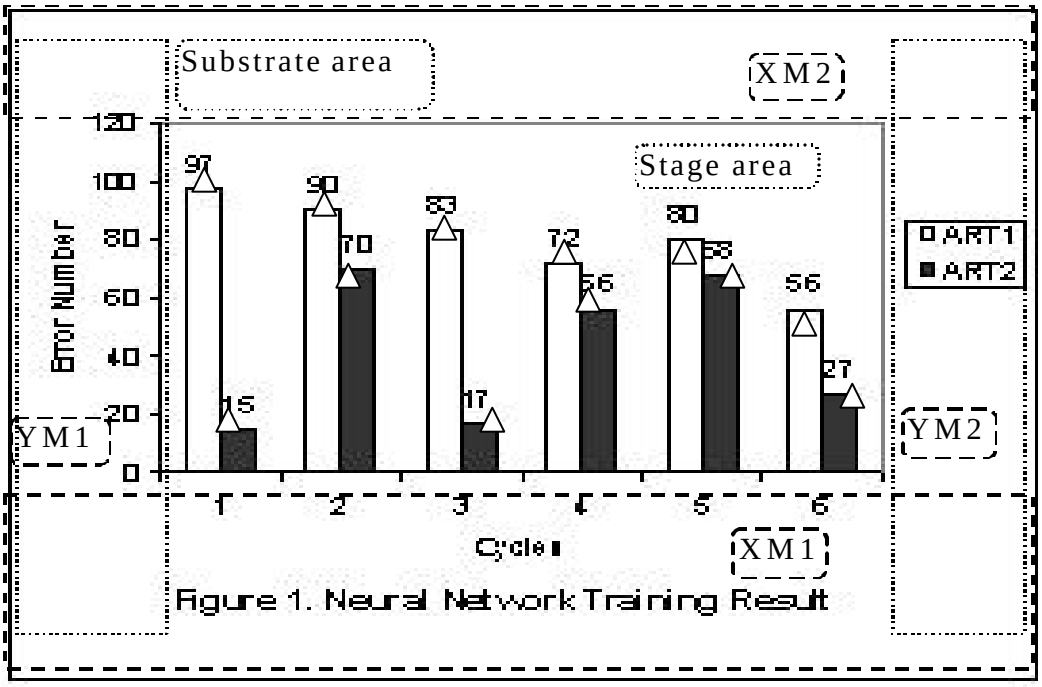


Figure 3.2: Definition illustrations in a two-dimensional-axes multiple-data-series chart. XM1: X-axis major area. XM2: X-axis minor area. YM1: Y-axis major area. YM2: Y-axis minor area. Note: areas are shown in the dashed rectangles. The names of the areas are in the dashed rectangles with oval corners. Δ: Value points.

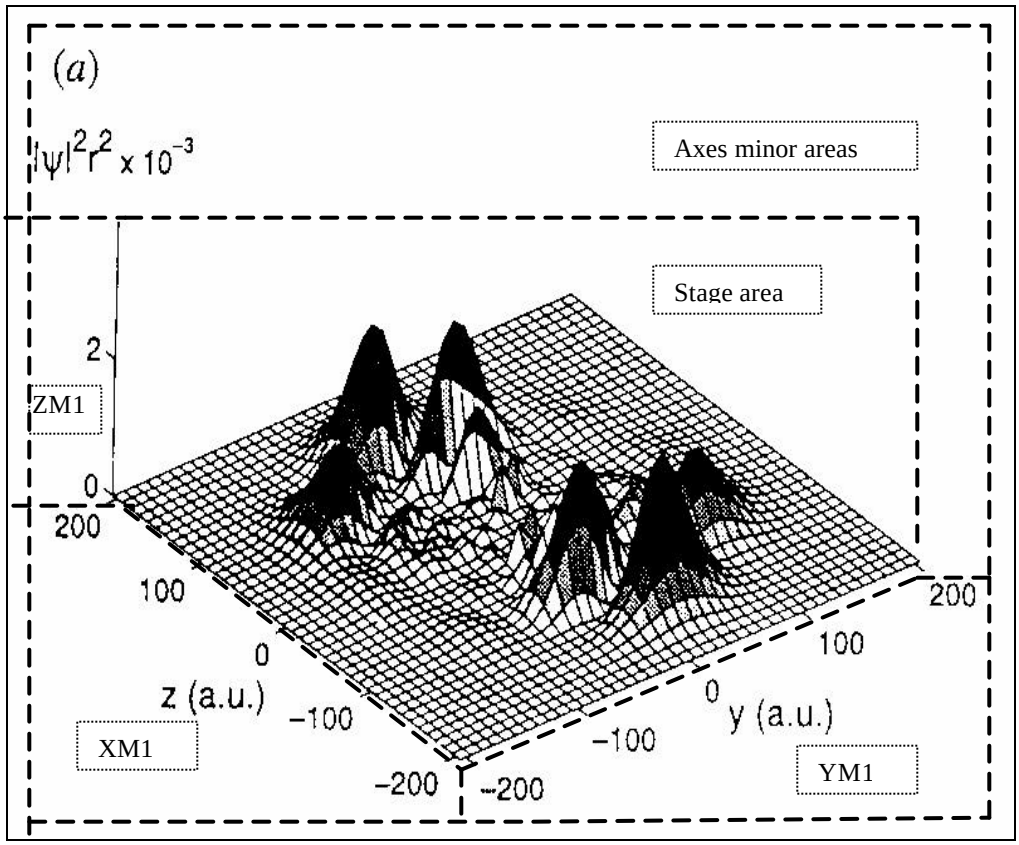


Figure 3.3: Areas defined in a three-dimensional-axes chart. XM1: X-axis major area. YM1: Y-axis major area. ZM1: Z-axis major area. Note: areas are shown in the enclosed dashed polygons. The names of the areas are in the dashed rectangles.

3.2 Methodology of Chart Recognition System

Our methodology of chart recognition system is inspired by what is known of human visual system.

3.2.1 Perceptual Organization on Charts

Chart creation tools tend to convert tabular data into an intuitive and vivid graphics representation using simple graphics primitive such as lines, curves, bars, etc. At the same time, however, the readers are less particular about the precision in a chart or diagram than they are with a table. Let us imagine the difference between a table of sales figures for a ten-year period and a line graph for that same data. We get a better sense of the overall trend in the graph at the expense of precise dollar amount. Unlike the table that presents quantitative data comparison by its rows and columns, charts present a qualitative and visual data comparison, such as trends and relationship.

Chart recognition which is a reverse processing of chart creation, attempts to convert the data in a chart or a graph into a tabular format. Since the data in a chart is in qualitative representation, the tabulated data recognized by the chart recognition tool may be not exactly the same as the original tabulated data that is used to create a chart. Once the tabular data is extracted by a chart recognition tool, some simple interpretation on the trends and the relationship about the data series, such as the maximum point, the minimum point, can also be inferred.

Psychologists suggest that human recognition of objects is guided by perceptual organization in the visual cortex. Perceptual organization refers to a basic capability of the human vision system to detect relevant groupings and structures from an image

without prior knowledge of its contents [100]. For instance, the human visual system has the ability to immediately detect symmetry, collinearity, covlinearity, parallelism, proximity and repetitive patterns when shown an arbitrary selected image. Perceptual organization actually can be categorized as the intermediate stages of vision [101]. It is above the low level sensory processing such as image enhancement, contrast normalization, but below the high-level recognition stage.

A chart image carries many properties lying in the perceptual organization level, such as parallelism, collinearity, repetition, etc. In figure 3.4, we generate an image (b) by cutting off the stage area in the chart image (a). We conducted a preliminary test involving a group of 10 subjects. All of the subjects were able to classify the image (b) as a chart within 2 seconds without any prior knowledge about image (a). The tick marker labels of the X-axis in image (b) are arranged on the same line. So are the labels of the Y-axis. These two lines are perpendicular to each other. Human visual system performs collinearity grouping on the text elements and relates the perpendicularity of the structural relationship to the chart structure. Graphics primitives in a chart also show the attributes in the perceptual organization level. Line and bar charts are the most frequent visualizations of data in general. Psychologists, Twersky *et al.*, viewed the bars as containers [115]. Each bar encloses one kind of things, separating that kind of thing from other kinds of thing, which may be in another bar. Lines, on the other hand, can be viewed as paths or connectors [115]. Thus psychologically, bars are used to convey discrete relationship, while lines are used to convey trends in a chart. The human visual system differentiates lines and bars by detecting different properties: collinearity and repetition, in the level of perceptual organization. In figure 3.5 (a), bars with the same pattern inside them are repeated with equal white space between each

other. The line in the figure 3.5 (b) shows collinearity that can be viewed as a trend. The tick markers along the X-axis of both images are short line segments that are parallel to each other and equally spaced between each other. So are the tick markers along the Y-axis. After obtaining these perceptual relationships, human visual system maps these tick markers to the structural elements of the charts at a high level recognition. Thus redundancy information is obtained from observing both text and graphics primitives. In general, redundancy reduces uncertainty.

After exploring human visual perceptual mechanism on recognizing a chart, we deduce the following three arguments for chart recognition.

Argument 1: Chart recognition is a hierarchical recognition procedure, starting from low-level vision, intermediate level vision to the final high-level interpretation.

Argument 2: The processing of the graphics part and textual part of a chart can be applied comparatively independent of each other.

Argument 3: The focus of chart recognition is at the intermediate level, i.e., the perceptual organization level.

Argument 1 is applicable to almost all the computer vision problems. Redundancy information in graphics and texts support argument 2. Nevertheless, feedback between them is helpful in the whole system. For instance, the orientation information from graphics processing is used in text processing in our system. Higher level recognitions such as recognizing the textures in bar patterns or recognizing the 3-D effect can provide more information about the charts, but are not the focus of the chart recognition problem.

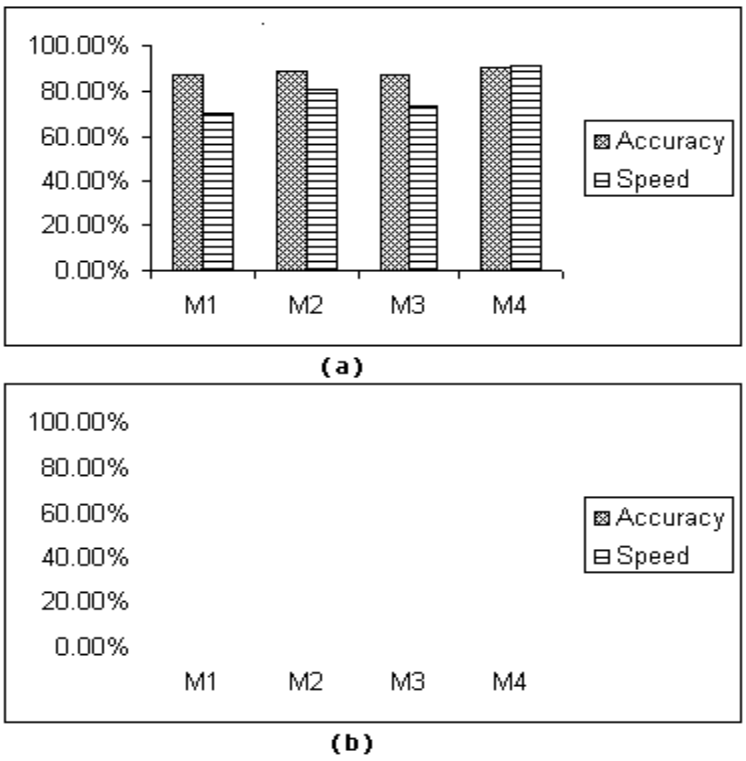


Figure 3.4: A perceptual test on a chart. (a) A testing chart image. (b) An image without the stage area of the testing image. All the ten subjects can categorize image (b) as a chart without the knowledge of image (a) within 2 seconds.

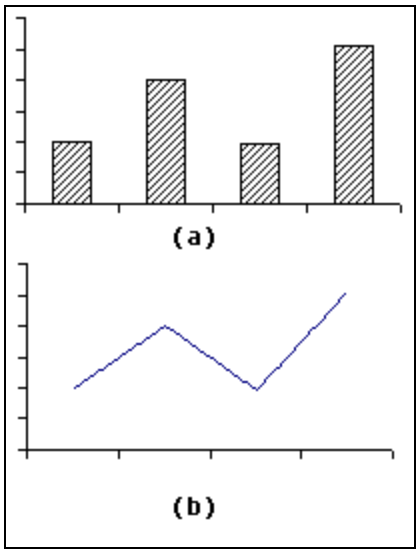


Figure 3.5: Graphic primitives in charts show properties of perceptual relationship. (a). Repetition of bar patterns. (b). Collinearity of a line segment

3.2.2 Methodology of the System

Motivated by the human visual system, the three arguments we deduce are the backbone of our chart recognition system.

The common graphics symbol processing sequence adopts a two-step processing: the first step is applied on the image level to get vectorized data. The second step processes the vectorized data to recognize the graphics symbol. Unlike this common sequence, our processing is a one-step processing, that is, the recognition of the graphics symbols is directly applied on the image level. One of the advantages for this methodology is that such sequence avoids the error caused by inaccurate vectorization which is also a noticeable problem in graphics recognition. Another advantage is that it sticks to the principle of focusing on the intermediate level processing that we deduce from human visual perception. We also argue that if the target of a particular recognition focuses on the lower level vision, one-step sequence is preferred to two-step sequence.

A hierarchical statistical-model-based chart recognition framework which focuses on the intermediate level of vision is proposed in our work. Figure 3.6 is the flow chart of our scientific chart recognition system. The framework comprises of three level processing: the low-level processing, the intermediate level processing and the high-level processing. The low-level processing which is also named preprocessing in our work performs four operations: binarization, noise removal, connected component analysis and connected component classification. The preprocessing techniques we adopt are all mature techniques. Brief introduction of the preprocessing is in section 3.3 of this chapter.

The intermediate level vision of chart recognition is the focus of our work. Image processing and computer vision techniques are first applied to extract those generally invariable graphics symbols such as the plot area and the axes. These works are reported in chapter 4. After the axes of a chart are detected, the areas defined in figures 3.2 and 3.3 can be segmented which also separate the graphics primitives and text primitives into comparable independent areas. Thus we can apply text processing and graphics processing independently in accordance with the second argument we have deduced. The information of detected axes is also used in dimension classification in which a chart is categorized as a 2-D chart or a 3-D chart.

Data markers inside the stage area are the highly variable graphics symbols that are the main cause of the great variety of charts. If image processing based primitive extraction methods are applied to extract data markers, such a system will be intractable and difficult to extend since the variation of the data markers is huge. Recognizing the detailed information about textures of the bar patterns and 3-D effect detection are not in line with the third argument. Besides, these topics remain as difficult topics in the computer vision area. Therefore in order to extract data markers efficiently and accurately without falling into the dilemma of texture recognition or 3-D object recognition, statistical models are proposed in this dissertation to model the repetition of data markers in the perceptual organization level. Four chart models are constructed and trained. They are the separated bar chart model, the contiguous bar chart model, the single-line-series chart model and the multiple-line-series chart model. Charts are classified by selecting the most probable model. The statistical modeling technique is also applied to segment data markers and extract the value points of them. The detailed work is reported in chapter 5.

The texts in a chart are structurally arranged along the graphics symbols such as the axes. To label and recognize these structured texts, a generalized X-Y tree structure named zoned directional X-Y tree is proposed in this dissertation to hierarchically represent the structured texts. A corresponding tree generation algorithm is also proposed to arrange the texts into a tree structure. Tick mark labels and titles are extracted from the zoned directional X-Y tree with syntactical analysis. Extracted words images are fed to the optical character recognition (OCR) module to get text results. In the high-level vision, the value points of each data series extracted from graphics processing are correlated with tick mark labels detected by text processing to generate a tabular interpretation for most 2-D charts. Tabular data are the output after correlating the value points with tick mark labels. Chapter 6 addresses issues of text primitives analysis and chart interpretation.

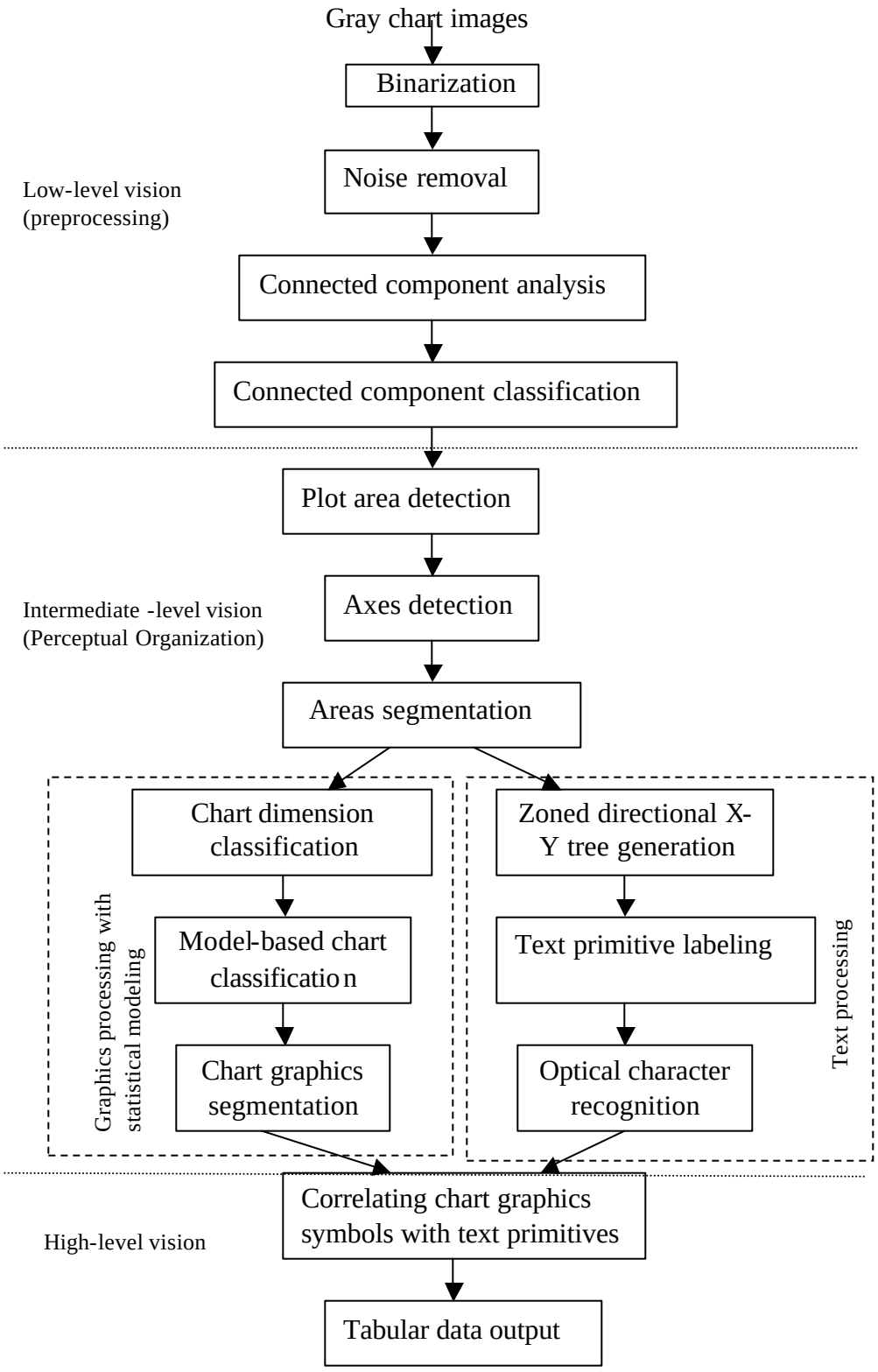


Figure 3.6: Flow chart of scientific chart recognition system.

3.2.3 System Assumptions

Chart recognition is a complex problem. We make the following simplifications and assumptions about the problem domain to research into various issues in chart recognition.

1. The chart images we process are in gray scale or binary format. Color gives more information to the chart images but may add complexity to our problems.
2. We assume that the background of chart area is monotonic, and not textured or halftoned. Extracting text or graphics from a textured or cluttered background is still a difficult topic.
3. We assume no gridlines along the X-axis or the Y-axis. The gridlines help the user to view the data easier. But in a recognition system, they make the background of the stage area more cluttered. Besides the gridline is not difficult to be removed by applying particular gridline removing procedures. Nevertheless, it should not add difficulties to our main topics.
4. No border lines enclose around the two axes. The border lines are the two lines connecting with the two axes to form the frame of the stage area. When the two axes are detected by the recognition system, the border lines can be easily removed by applying a border line removal algorithm in the future work.
5. The reading order of the chart is from left to right. The reading orders can be from top to down, or from bottom to up, or from right to left. Nevertheless, the reading order from left to right is the main order among most charts.

3.2.4 Testing Data Collection

Currently known document image analysis databases such as NIST databases for document image analysis or UW document image databases I, II and III [89], are not suitable for our work since they contain few chart images. Our testing chart images were collected by scanning charts on technical journals at a resolution of 300 dpi. The technical journals were arbitrarily selected, such as *IEEE Transaction on Pattern Analysis and Machine Intelligence, Journal of Pattern Recognition, Journal of Fish, Journal of Physics* etc. The chart images were first photocopied and then scanned. Thus noise and discontinuity caused by low tone may be introduced. No strict restriction or alignment on the scanner bed is required. Thus the chart images may be skewed when photocopied or scanned. The skew degrees for most of the scanned charts are within $\pm 5^{\circ}$. Table 3.1 shows the number of different categories of charts as testing data.

3-D charts	2-D charts			Total
	Bar charts	Line charts	Other charts	
44	254	207	32	537

Table 3.1: Testing data distribution of chart recognition system

A total of 537 chart images were collected, including 44 3-D chart images and 493 2-D chart images. Among 493 2-D charts, there are 254 bar charts, 207 line charts and 32 other charts. Bar charts are grouped into one-bar-series charts and multiple-bar-series charts. The textures inside the bar patterns are not fixed since the images are acquired by arbitrarily selecting from the journals. The growing directions for the bar patterns are up, down, or mixed. For most charts, the growing directions are up. Likewise, line charts are

classified as one-line-series charts and multiple-line-series charts. These two categories of line charts can be further classified into a with-marker group and a without-marker group. The other charts include data charts, high-low-close charts, etc.

3.3 Preprocessing

Four kinds of preprocessing operations are applied on the original images: binarization, noise removal, connected component analysis and connected component classification.

Binarization

The chart images may be binarized or in gray level. The gray images first need to be binarized. There are a lot of binarization algorithms, such as p-tile, optimal thresholding, adaptive thresholding, etc [60, 111]. We adopt the thresholding technique proposed by Otsu [87].

Noise Removal

After the image is binarized, a simple noise removal morphological operation is applied on the binary image to remove isolated noises. The isolated noises are dot noises whose eight neighboring pixels are all background pixels. A 3 by 3 all-ones structural element is applied on the images. The pixel is considered as a foreground pixel if its value is greater than one.

Connected Component Analysis

Connected component analysis is a mature technique. In our method, we assume the foreground and background to be black and white respectively. In this technique, we use eight-connectivity in the foreground and four-connectivity in the background to group pixels into distinctive components. The connected component analysis we adopt is sequential connected components algorithm in [52]. The output of this operation is all the labeled components. Each component has a feature vector to identify its properties. We calculate seven kinds of properties: labels of connected components, top and bottom points position of the bounding box, mass, centroid position, height of the bounding box, width of the bounding box, and density. The density means the ratio of mass to the area of the bounding box.

Connected Component Classification

In this procedure, we classify the connected components into graphics-like components or character-like components. In the connected components set, the height and width of graphics elements are usually larger than those of text elements. Even though some text elements with large font size may have a large height and width value, the ratio of mass to height or width is larger than that of graphic elements. In our method, we use three filters for classification. These three filters are height filter, width filter, and density filter. We apply a histogram technique to get the threshold for each filter. Those components whose height and width exceed the corresponding thresholds are labeled as graphics-like elements. If the height, the width and density of a component are less than the respective thresholds, it is classified as a character-like component. Otherwise, it is classified as a graphics-like component.

3.4 Summary

In this chapter, we have defined notational conventions of a scientific chart both from a generation point of view and recognition point of view. The methodology of our chart recognition system that tries to mimic the human chart recognition process is also presented. The basic constraints and assumptions of our recognition system are also presented. Testing data are introduced. Four preprocessing steps in our system: binarization, noise removal, connected component analysis and connected component classification, are introduced in this chapter.

The contributions of this chapter are as follows:

1. We propose relevant notational definitions for scientific charts from a recognition point of view.
2. We give theoretical contributions in constructing a chart recognition system inspired by what is known of human visual system. The three arguments we deduced are the principles and backbone of our chart recognition problems.
3. A hierarchical statistical-model-based chart recognition framework which focuses on the intermediate level of vision is proposed in this dissertation.
4. Testing data set up. The procedure of setting up the testing data for our system is not complex but tedious.

Chapter 4

Chart Graphics Symbol Recognition

The procedure that follows the pre-processing procedure is the chart graphics symbol recognition. Our symbol recognition process is a one-stage process, that is, image processing operates directly on the image. Prominent and comparatively invariant graphics symbols such as the coordinate system of the charts are detected directly from the pixel level so as to avoid the errors caused by vectorization. In order to recognize the graphics symbols in a chart, the plot area of a chart where most graphics elements are active must be detected correctly first. Our graphics processing at this level includes plot area detection, chart axes detection, bar pattern extraction. Chart axes detection is to detect the position of the coordinate system of the testing chart and from the obtained dimension information to classify the chart into the two-dimensional-axes chart or the three-dimensional-axes chart. After obtaining the coordinate system, we then divide the image space into graphics primitive division and text primitive division. In the next section, plot area detection will be proposed first.

4.1 Plot Area Detection

The plot area is the area bounded by the axes within which all the main data markers lie. Normally, the index or the tick names are very close to the corresponding axes. Plot area detection is to mark out an area that includes at least all the elements inside the plot area and overlaps the plot area as much as possible. Suppose for the actual plot area *A*, the area detected by the detection algorithm is *B*. Then the relationship between *A* and *B* is that *B* includes or equals to *A*. The plot area detection is not complex since we assume that the chart has already been detected by page layout analysis. But this process is an important start for a robust chart recognition system. Without correctly locating the plot area, any further processing is in vain.

Plot area detection is not explicitly presented in both Yokokura and Watanabe [121] and our previous bar chart recognition algorithms [126, 128]. Both of the algorithms attempted to find the area enclosed by the X-axis and Y-axis for further graphics primitive extraction. We name our previous method as Zhou_old method [126, 128], Yokokura and Watanabe's method as Yokokura's method and our present plot area detection method as Zhou_new method.

Yokokura's plot area detection method is such that: first, generate the vertical and horizontal projection profiles for the chart image. Second extract the left-most peak and the lowest peak in the vertical and horizontal projections respectively and label them as the Y-axis and the X-axis respectively. Third, label the upper-right zone divided by the Y-axis and the X-axis as the plot area.

Zhou_old plot area detection method works as follows: first, apply connected component analysis on the whole chart image. Next, find the component with the largest area. The area bounded by the rectangle of it is set as the plot area.

Unlike Zhou_old method which is a bottom-up method, Zhou_new method is a top-down method which also utilizes the projection profiles of the image. Figure 4.1 illustrates the steps of Zhou_new method. Zhou_new method is like applying X-Y cut algorithm only on the whole image level. The experiments and analysis for the three methods will be discussed in section 4.3.1.

1. Generate the horizontal projection for the whole image.
2. Get the horizontal segments by cutting the horizontal projection.
3. Merge the neighboring horizontal segments with distance less than T_h.
4. Find the largest horizontal segment and get the horizontal image strip from it.
5. Generate the vertical projection for the horizontal image strip.
6. Get the vertical segments by cutting the vertical projection.
7. Merge the neighboring vertical segments with distance less than T_v.
8. Find the largest vertical segment and set the rectangle of it as the plot area.

Figure 4.1: Algorithm of Plot Area Detection.

4.2 Chart Axes Detection

After detecting the plot area, the X-axis and the Y-axis, or the Z-axis if it is a 3-D chart, should be extracted for the subsequent decomposition. Two approaches are used to detect chart axes: projection-based and Hough-based.

4.2.1 Projection-based Axes Detection

Yokokura's plot area detection method already detected the X-axis and Y-axis when it found a plot area. Zhou_old plot area detection assumes the leftmost peak and the lowest bottom peak in the plot area as the Y-axis and the X-axis respectively. Since the projection-based detection algorithm is computationally fast, we propose an improved projection-based axes detection method, named Zhou_Axis_projection method. The Zhou_Axis_projection method first detects the peaks in the vertical and horizontal projections. Then it obtains the leftmost position in the vertical projection and the lowest position in the horizontal projection that are greater than a third of the peaks respectively as the positions of the X-axis and Y-axis on the plot areas.

Projection-based axes detection algorithms obtain the approximate positions of axes lying in the horizontal and the vertical directions. They cannot correctly detect the axes lying in other different angles than vertical or horizontal directions such as in the case the axes are distorted or skewed seriously for some scanned chart images. They cannot be applied successfully to detect the axes in 3-D charts either. In order to obtain robust result of axes detection for further chart recognition, we propose a novel Hough-based axes detection algorithm with geometric analysis to detect the axis lying in an arbitrary direction. This is explained in detail in the ensuing section 4.2.2.

4.2.2 Hough-Based Axes Detection with Geometric Analysis

Hough transform is a voting technique to detect straight lines by utilizing a parametric equation to transform each point in the image to be processed from Cartesian feature space (X-Y) to a new parameter space (r-q). We adopt the following polar parameterization for a straight line.

$$r = x\cos q + y\sin q \tag{4.1}$$

The standard Hough transform algorithm can be found in [52]. We also propose a concise and algebra representation for it in Appendix A.

The axes detection algorithm can be outlined as four procedures:

- Preprocessing and applying Hough transform.
- Multi-section peak clustering in Hough domain.
- Line segment verification in the image domain and merging.
- Geometric axes reconstruction.

Figure 4.2 shows the main step of Hough-based axes detection algorithm with geometric analysis. We describe the details of the four procedures in the following sections.

1. Preprocess the plot area and apply Hough transform.
2. For each q section in the Hough space, do step 3 to step 8 to obtain the peak information.
3. Binarize the accumulator section.
4. Clustering the peaks to obtain the peak objects list in the section.
5. Calculate the value for the peak objects.
6. Sort the peak objects along the r and check related peak pairs for the neighboring peak objects.
7. Get the peak objects with the maximum and minimum r.
8. Verify each peak object in the image space for valid line segments. The length of the valid line segments is larger than one fifth of the longest line segment found in that section.
9. Merge the line segments in the image space.
10. Detect the vertical axis by selecting the leftmost line segment whose end point is closest to the left boundary of the plot area in the vertical major section.
11. Find the line segment whose end point is closest to the bottom boundary of the plot area and verify it. If it is the second axis in a two-dimensional chart, stop the program, else go to step 12.
12. Find the third axis which has shortest end point distance with the found axes.

Figure 4.2: Hough-based Axes Detection Algorithm with geometric analysis

Preprocessing and Applying Hough Transform

For a large number of feature points, the Hough transform is computationally expensive hence a preprocessing on the detected plot area should be applied first. In our preprocessing, we first filter the character-like elements inside the plot area. Thus a lot of feature points are dropped. Then we get the interior boundary image for the remaining graphic image as feature points for Hough transform using the boundary generation algorithm in [95]. The points inside the solid bars are also dropped, which minimize greatly the number of voting points. Therefore the number of feature points for voting is small. The resolution along the q is set to one degree. The resolution along the r is set to one degree too. Both the increment step of r and pixel in feature space are the same in the vertical and horizontal direction. Thus we can get a close correlation between the Hough space and the image space using this r resolution.

Multi-Section Peak Clustering in Hough Domain

The peaks in the Hough space represent the straight lines. In order to delineate the axes, more detailed processing on the result of Hough transform should be implemented. We first divide the q into 18 sections: starting from the division combining $175^\circ \sim 180^\circ$ and $0^\circ \sim 5^\circ$, then $6^\circ \sim 15^\circ$, until the division $165^\circ \sim 174^\circ$. The two sections where the horizontal and vertical lines lie are called the major sections. The other sections are called minor sections. The resolution of quantization of the Hough space is important for correctly detecting straight lines in the Hough space. If the resolution is too small, there will be over-fitting problem. In order to avoid the spurious peaks caused by over-fitting, a peak points clustering procedure is applied on the peak points. In each section, the peak

points are first projected in the horizontal direction. The horizontal profile is segmented into *peak objects* with an allowable discontinuity distance. The peak objects in the sections usually comprise of multiple bins. Suppose there are M bins composing of a peak object. For a specific bin inside the peak object, the value of the bin is $H_l(i, j)$. The value of the peak object is computed by

$$V_{peak} = \frac{\sum_{l=1}^{M} H_l(i, j)}{M} \tag{4.2}$$

For the neighboring peak objects along the r direction, if the value difference between them is less than one tenth of the smaller peak value, these two neighboring peak objects are considered as a pair of *related peak objects.*

Line Segment Verification in the Image Domain and Merging

Hough transform renders the approximate positions of the straight lines. Some of results from Hough transform may represent a cluster of dashed line segments or points. In order to detect axes, a line segment detector, a line segment verification procedure, is first applied in the each section to find valid line segments which are above a threshold T_{Axis1} from the voting image pixel list. Then line segments on the same straight line that the internal distance is less than T_{Inter} are merged. The inter-section line merging is also applied to overcome the problem of duplicated representation of a line segment.

Geometric Axes Reconstruction

We first analyze the geometry of the axes in both two-dimensional-axes and three-dimensional-axes charts. The normal reading order is from left to right. Figure 4.3 illustrates the geometry of the axes. In reconstructing the axes, the leftmost line segment in the major vertical section is selected as the Y-axis in two-dimensional-axis charts or Z-axis in three-dimensional-axis charts. The line segment of which one end point (x_1, y_1) is closest to the lowest bottom of the plot area is the candidate of axes. If this line segment lies in the major sections, then it is the X-axis and the chart is classified as the two-dimensional chart. If it lies in minor sections, then it may be the X-axis or the Y-axis in a three-dimensional chart as shown in figure 4.3 (b). If y_c of the crossing point (x_c, y_c) is smaller than y_1, then the second detected axis is the X-axis, otherwise it is recognized as the Y-axis. The remaining axis is found by searching the shortest distance between the end points of the second found axis and other line segments in the Hough space. The crossing point of the Y-axis with the X-axis in a two-dimensional chart or the crossing point of the Z-axis with the X-axis in a three-dimensional chart is the origin point (x_o, y_o) of the coordinate system. Experiments and result analysis will be reported in section 4.3.2.

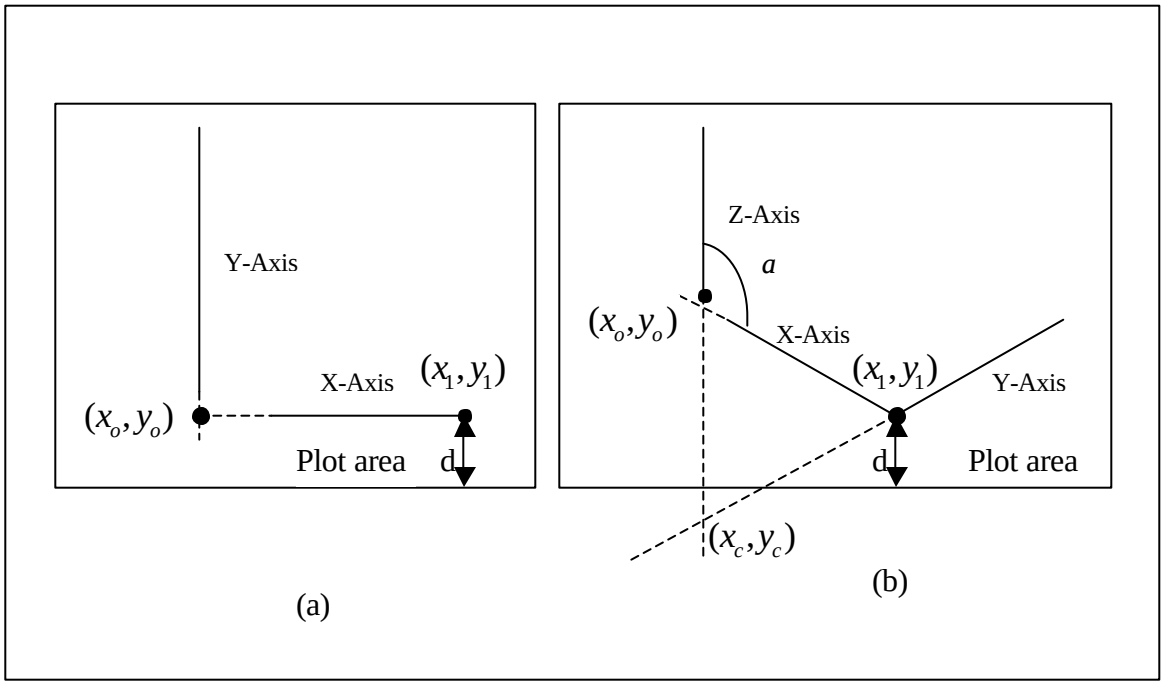


Figure 4.3: Geometry illustration of the axes in charts. (a) X-axis and Y-axis in the two-dimensional chart. (b) Z-axis, X-axis and Y-axis in the three-dimensional chart. Note: the solid line segments are recovered line segments from the Hough space.

4.3 Experiments and Analysis

In this section, we present experiments on the algorithms in detecting plot areas and axes which are proposed in previous sections. The output of the plot area detection is a rectangle that includes the plot area. We draw the detected plot areas directly on the testing image by using red rectangles. The outputs of axes detection are the coordinates of two end points of line segments. We reconstruct the axes on the testing images by drawing them in red color. The origin of the coordinate system is represented in blue color. The topics on performance analysis and benchmark development are incurring more and more interests in the document image analysis domain [43, 70, 90]. Developing benchmarks to evaluate the performance will be future work. To evaluate the

performance of the algorithms in our work, we adopt two evaluation criteria, precision and recall from the information retrieval domain [94]. The *precision* of the algorithm is the proportion of detected objects that are actually correct. The *recall* is the proportion of correctly detected objects to all actual objects present in the chart.

4.3.1 Results of Plot Area Detection

537 chart images that have 537 plot areas in total were used to test the three plot area detection methods discussed earlier. Among these 537 chart images, there are 493 images with 2-D axes and 44 images with 3-D axes. The thresholds in Zhou_new method are: $T_h = 4$ and $T_v = 7$ respectively. Both thresholds are fixed for all tested images. Figure 4.4 shows an example that all of the methods succeed in finding the plot area. Table 4.1 illustrates the testing results of all the three plot area detection methods. Since all of the three methods render a plot area for each testing image, the recall and precision are equal. Therefore table 4.1 only shows the results of recall. Recall measures the proportion of correct plot areas to all the plot areas.

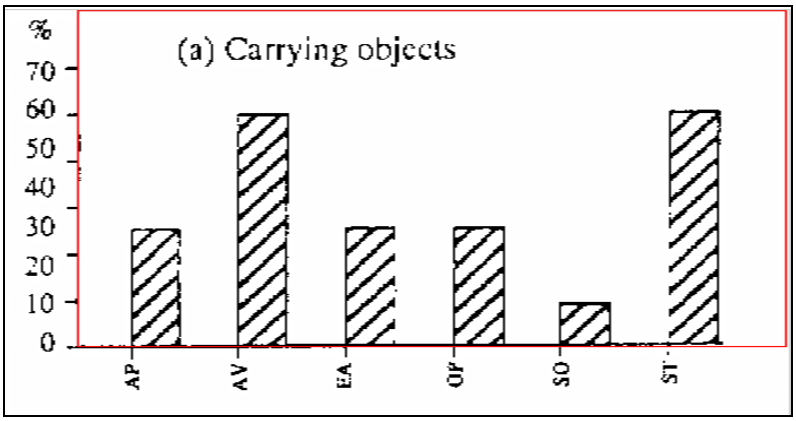
Methods	2-D plot areas (total 493)		3-D plot areas (total 44)		Total plot areas (537)	
	Correct area detected	Recall	Correct area detected	Recall	Correct area detected	Recall
Yokokura	237	48.07%	0	0%	237	44.13%
Zhou_old	386	78.30%	38	86.36%	424	78.96%
Zhou_new	475	96.35%	44	100%	519	96.65%

Table 4.1: Testing results of plot area detection methods

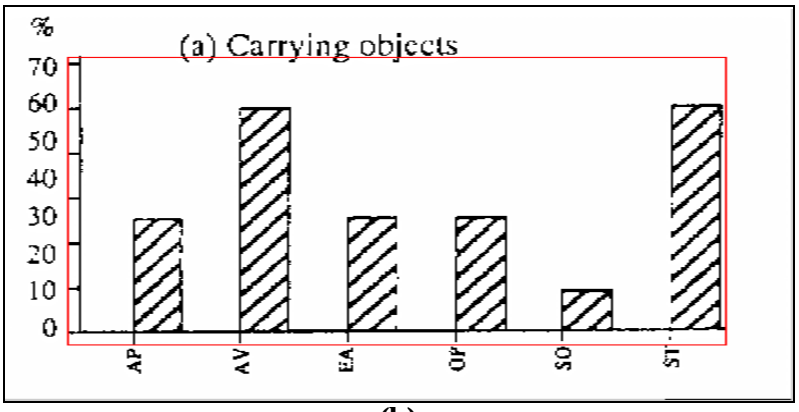
From the above result table, we see the Zhou_new method renders the best detection rate both on 2-D and 3-D charts. The performance of Yokokura method is the worst. The performance of Zhou_old method has a great improvement comparing with that of Yokokura method, but still worse than the performance of Zhou_new method.

Figure 4.5 presents the wrong plot area detection examples by the three methods. The plot areas in subfigures (a) to (g) of figure 4.5 are wrongly detected by both Yokokura and Zhou_old methods. Yet Zhou_new method can detect all of them correctly. All the above methods fail in detecting the plot areas of subfigures (h) and (i) of figure 4.5. Figure 4.5 only shows the results of one of the algorithms that detect the plot areas wrongly. Although 98% of the Y axes and X axes of the testing 2-D charts lie in the left and bottom part of the images respectively, the detection rate of Yokokura's method is not satisfactory, not to mention its result on the testing 3-D charts. On visual inspection, we note that the axes are the two salient peaks in a chart. But in a projection profile, several bins represent the position of an axis due to the non-smoothness of the line, noise, or other uncertainty effect. Other graphic elements such as a solid bar may compete with the axis as in subfigures (a) and (d). Other problems such as interference by the outside frame of a plot area or the skewness of the axes or even the whole charts like the subfigures (b) and (d) may also affect the precision of plot area detection. Therefore, using a naive observation to treat the upper-right area divided by the leftmost peak and the lowest peak of vertical and horizontal projections as the plot area, is not robust for a practical chart recognition system. The problem degrades the accuracy of Zhou_old plot area detection most is caused by broken graphics elements, especially when the axes are severely broken due to scanning or other reasons as in subfigure (e) and (f). Even if there

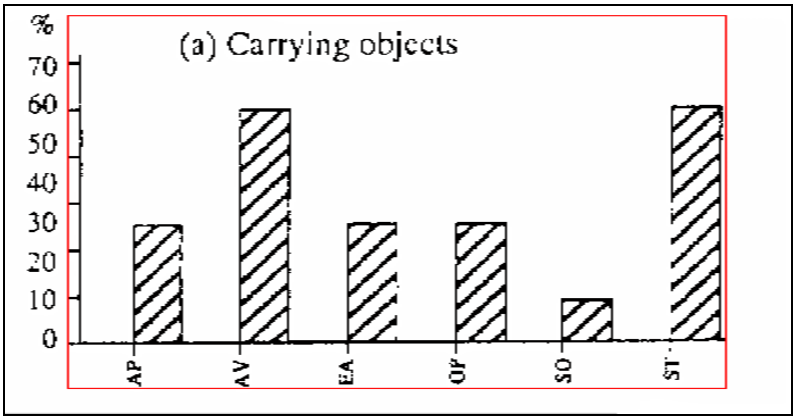
is no broken graphic element, a non-axis graphics element such as in subfigure (g) may also be mistaken as the plot area due to its larger size. Zhou_new method lessens the major problem of Zhou_old method by adding a merger operation. Nevertheless, same problems remain as in subfigure (h) and (i) if the merging threshold T_v is not suitable for these images.



(a)

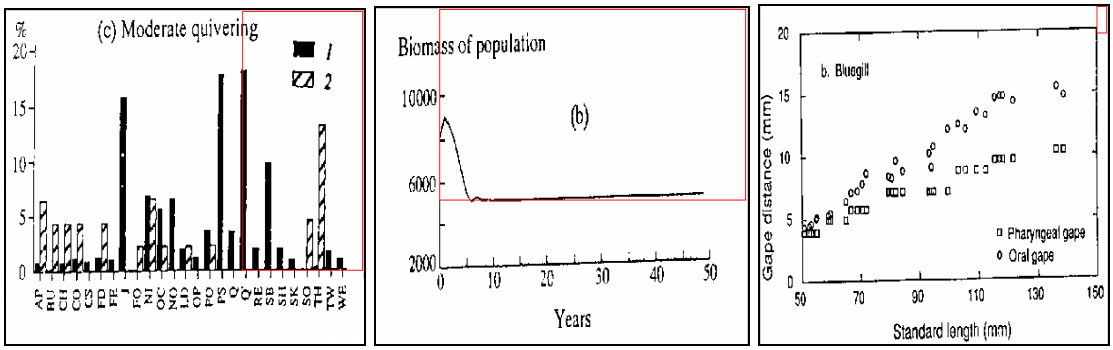


(b)

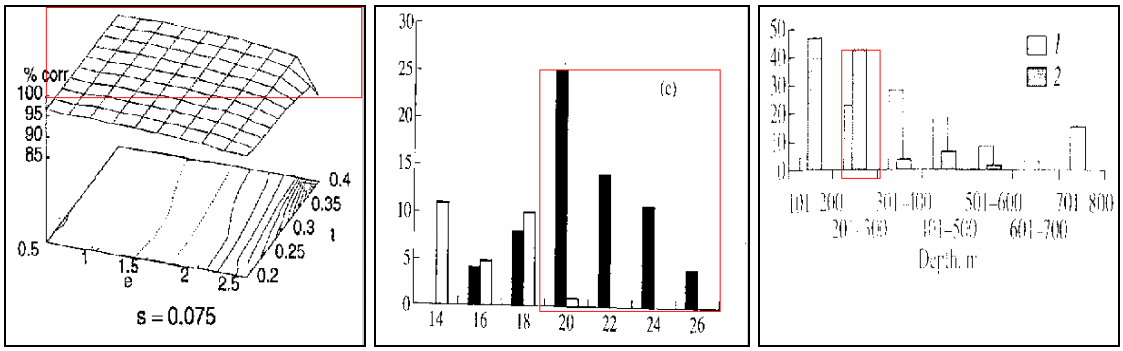


(c)

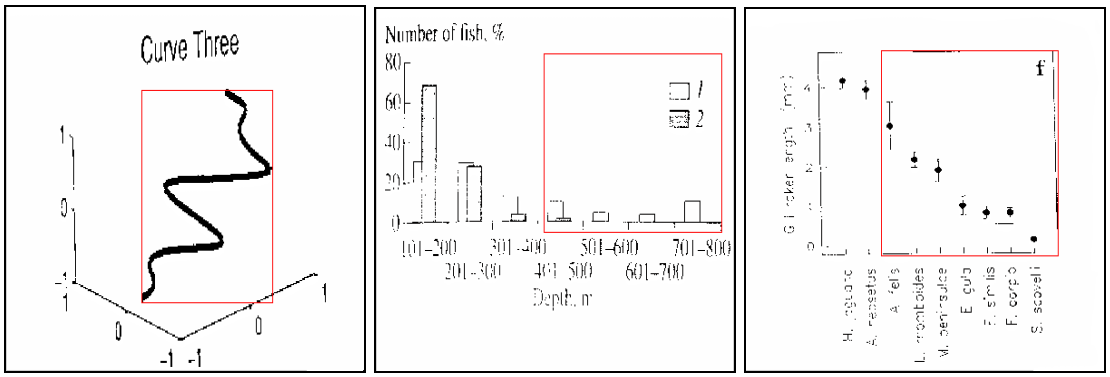
Figure 4.4: The results of plot area detection on an example image: (a) the result of Yokokura method. (b) the result of Zhou_old method. (c) the result of Zhou_new method. Note: the red rectangles are detected plot areas.



(a) **(b)** **(c)**



(d) **(e)** **(f)**



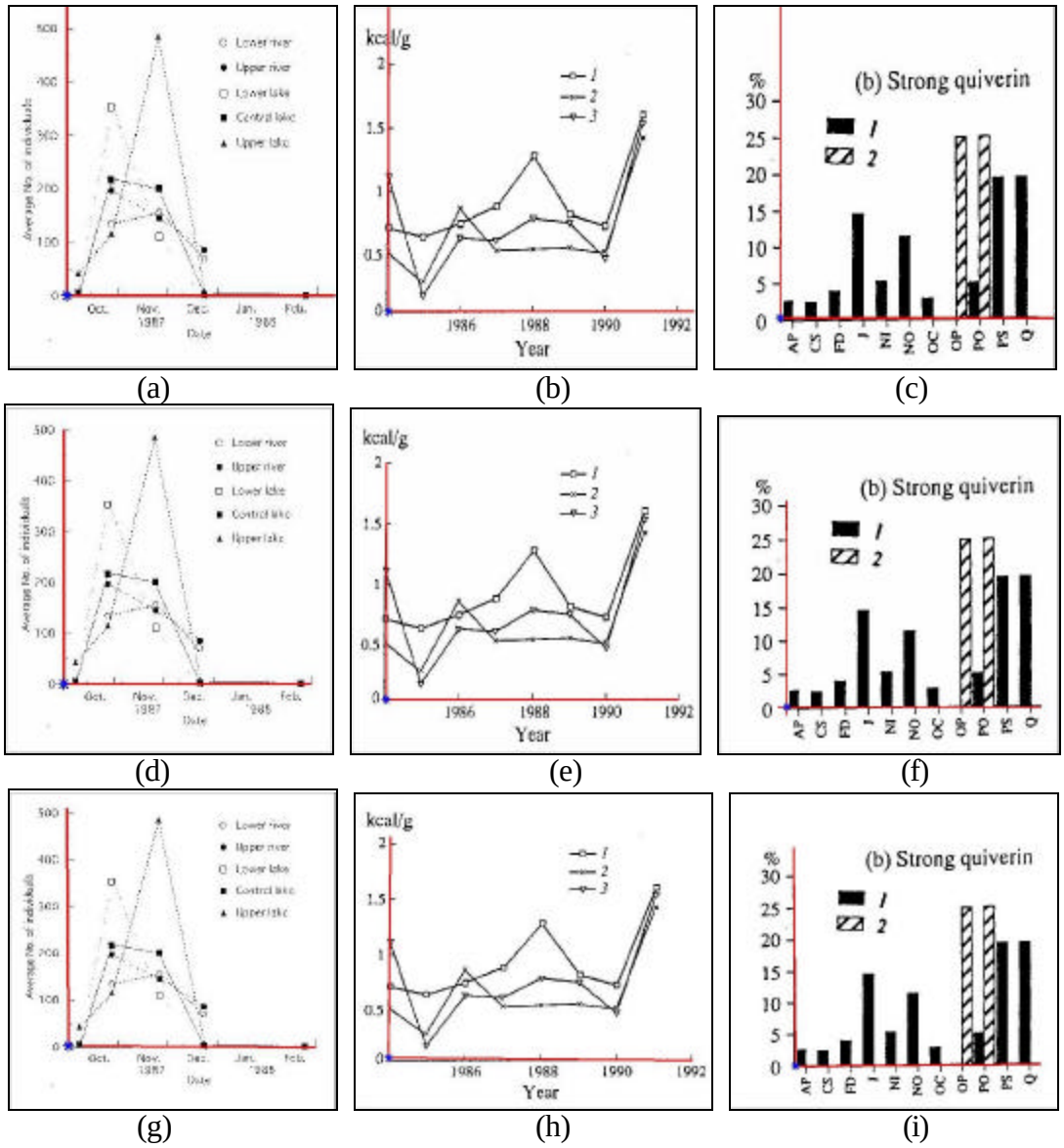
(g) **(h)** **(i)**

Figure 4.5: Wrong detection results of plot area detection: (a)—(d) wrong results of Yokokura method. (e)—(g) wrong results of Zhou_old method. (h)—(i) wrong results of Zhou_new method. Note: the rectangles are detected plot areas.

4.3.2 Results of Chart Axes Detection

Projection-based axes detection methods in [121, 127] are also implemented to compare the results of Hough-based axes detection algorithms. Two projection-based algorithms are implemented: Yokokura method and Zhou_old method. Figure 4.6 shows some successful examples of axes detection of the three algorithms. The red lines depict the position of X-axis and Y-axis respectively and the blue asterisk point indicates the origin of the coordinate system. Besides detecting the axes lying in the ordinary orientation, Hough-based algorithm can detect axes that both projection-based algorithms cannot find such as three-dimensional coordinate systems and also slant two-dimensional coordinate systems, or even hand-drawn charts as shown in the figure 4.7. We name Hough-based algorithm as Zhou_new method. The thresholds in Zhou_new method are: $T_{Inter} = 7$ and T_{Axis1} equals to one tenth of the largest peak. Both thresholds are fixed for all tested images.

Table 4.2 gives the result of comparison for the three algorithms in testing on 2-D charts. Table 4.3 shows the result of comparison for the three algorithms in detecting axes of 3-D charts. Precision is defined the proportion of detected axes that is actually correct. Recall measures the probability of correct detected axes on existing axes. The performance of detecting each axis in both two-dimensional and three-dimensional coordinate systems is presented in detail. Projection-based methods cannot detect the X-axis and the Y-axis in three-dimensional charts and also perform poorly on detecting the vertical axis. However both methods will always output an X-axis and a Y-axis for each testing image but both precision and recall for Y-axis detection is obviously lower



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 4.6: Successful examples of axes detection algorithms: (a)—(c) results from Yokokura method, (d)—(f) results from Zhou_old method, (g)—(i) results from Hough-based method. Note: the red line segments are axes and the blue points are the origin of the coordinate systems.

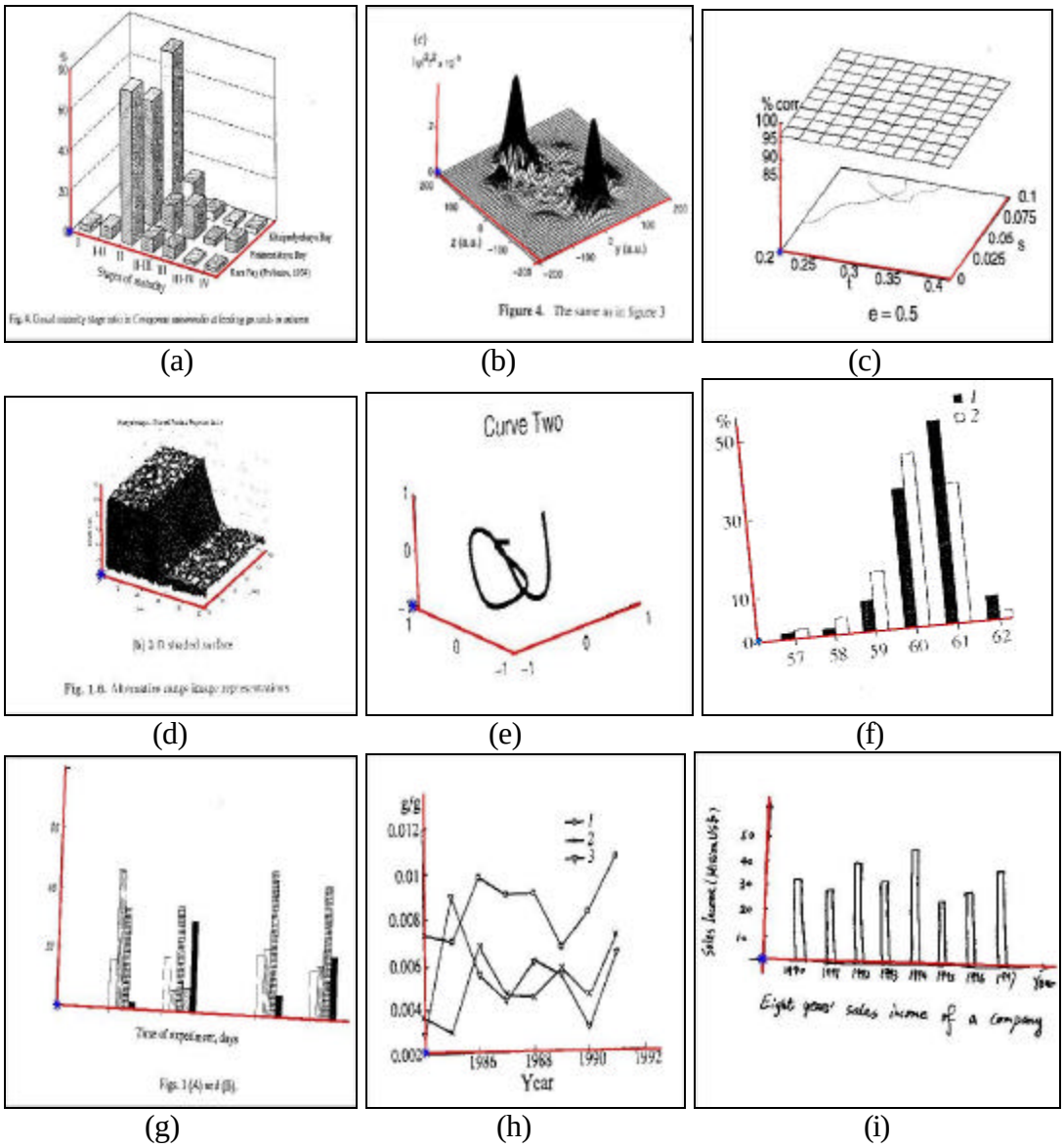


Figure 4.7: Successful results of our Hough-based axes detection algorithms. The other two methods failed on these images. (a)—(e) three-dimension-axis charts, (f)—(h) two-dimension-axis charts from technical journals, (i) a two-dimension-axis hand-draw chart. Note: the red line segments are axes and the blue points are the original point of the coordinate systems.

than those of X-axis detection. One of the major reasons is the interference of the graphic primitives such as solid bars, etc. Hough-based algorithm outperforms greatly in all the axes detection for both two-dimensional and three-dimensional charts shown from table 4.2 and table 4.3.

Hough-based detection algorithm shows comparatively robust performance in detecting two-dimensional axes on performance of both precision and recall. By examining the display of axes output, we also find that the position and orientation of the correctly detected axes from Hough-based algorithms are better than those from projection-based methods when the real axes in the testing chart have some skew. The performance on detecting the Z-axis of the three-dimensional coordinate system is also robust. However many Y-axes in three-dimensional charts are missing, lowerimg the recall rate. Figure 4.8 (e) and (f) illustrate two examples of such a case. The missing axis is caused by interference of other graphic primitives. Nevertheless, they still can be classified as a three-dimensional chart since the clockwise angle between Z-axis and X-axis is greater than100°.

Although Hough-based axes detection algorithm shows satisfactory results on detecting the two-dimensional axes, there are cases where it fails in detection as illustrated in Figure 4.8. One of the main reasons is the propagation error caused by the plot area detection. For example in figure 4.8 (a), an incomplete plot area leads to a wrongly detected Y-axis while in figure 4.8 (b) the Y-axis is totally missing. Although the Hough-based method is tolerant to the discontinuities of line segments, if the quality of image is poor and broken lines are widely segmented, other data markers or graphic

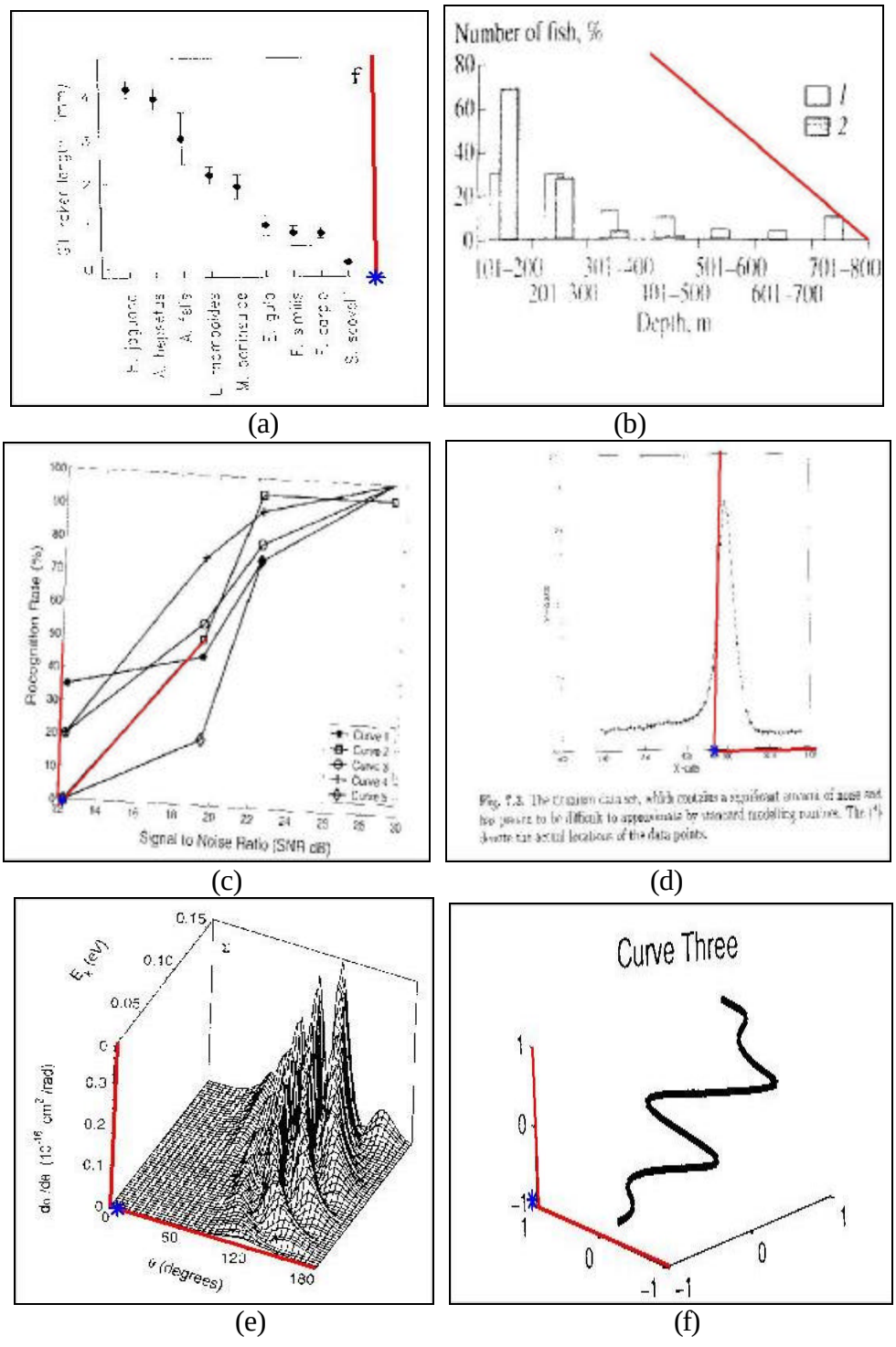
elements in the plot area may prevail over the real axes and hence are erroneously selected as the axes like in figure 4.8 (c) and (d).

Categories / Methods	Detected Axes		Correct Axes		Precision (%)		Recall (%)	
	X-axis	Y-axis	X-axis	Y-axis	X-axis	Y-axis	X-axis	Y-axis
Yokokura	493	493	323	240	65.52%	48.68%	65.52%	48.68%
Zhou_old	493	493	413	387	83.37%	74.94%	83.37%	74.94%
Zhou_new	477	484	462	467	96.86%	96.49%	93.71%	94.73%

Table 4.2: Testing results of axes detection algorithms for 2-D charts

	Detected Axes			Correct Axes			Precision (%)			Recall (%)		
	X-axis	Y-axis	Z-axis	X-axis	Y-axis	Z-axis	X-axis	Y-axis	Z-axis	X-axis	Y-axis	Z-axis
Yokokura	44	0	44	0	0	15	0	0	34.1	0	0	34.1
Zhou_old	44	0	44	0	0	22	0	0	50	0	0	50
Zhou_new	43	32	44	37	29	43	86.05	90.63	97.73	84.09	65.91	97.73

Table 4.3: Testing results of axes detection algorithms for 3-D charts



(a) (b) (c) (d) (e) (f)

Figure 4.8: Unsuccessful results by Hough-based axes detection algorithm: (a)—(d) two-dimension-axis charts, (e)—(f) three-dimension-axis charts. Note: the red line segments are axes and the blue points are the origin of the coordinate systems.

4.4 Summary

In this chapter, we propose our processing on the intermediate level chart symbol recognition. First, the plot area is detected by applying an improved projection-based algorithm. Next, a novel axes detection algorithm is further applied on the plot area to detect the coordinate system. Experiments and performance analysis on the plot area detection and axes detection are illustrated.

The contributions of this chapter include:

1. We propose an improved projection-based approach for plot area detection.
2. We present a method of axes detection with Hough feature clustering and geometric analysis in our work to detect 2-D and 3-D axes. The axes information will be used to classify charts into 2-D charts and 3-D charts in the stage of chart classification. Our experiment shows that the proposed axes detection is robust comparing with other projection-based methods.

Chapter 5

Chart Classification and Segmentation

In the previous chapter, we have investigated two graphical processing procedures, plot area detection and axes detection, which give us some essential information of a processed chart. Yet the information is still insufficient to enable us to interpret the generated tabular data since the data markers have not been extracted. The series of data markers are the key features representing the information conveyed by a chart. When generating a chart from a data sheet or a table, the attribute variation of the data markers such as height or area, represents the tabular data. The value points of the data markers grasp the attribute variation of the data markers. Thus segmenting the data markers and extracting the value points is the right path to interpret a chart.

Chart segmentation in our work is to segment the data markers and extract the value points. The diversity of charts is mainly due to the various types of data markers. Different data markers use different semantics to convey the underlying tabular data. Therefore before chart segmentation, *chart classification* should be applied in which charts are classified into their corresponding categories in terms of their shapes and semantics representations. Chart classification and chart segmentation interleave with

each other to some extent. For instance, after extracting the bar patterns or line patterns from a chart, we can classify the chart into a bar chart or a line chart accordingly. On the other hand, if a chart belongs to a specific class of charts, the knowledge of this class also helps to extract the data markers.

Chart classification in our current work addresses two problems: one is dimension classification to classify a chart into a 2-D chart or a 3-D chart, the other is type classification to categorize a chart into one of the four types of 2-D charts: separated bar charts, contiguous bar charts, single-line-series charts and multiple-line-series charts. Chart segmentation is to obtain the value points of the abovementioned four types of 2-D charts.

The topic of chart classification has not been very much researched in the area of chart recognition. Currently reported research works in [121, 126] all assumed that the type of the recognizing chart is already known such as a bar chart. Even with this assumption, the algorithm they proposed could only apply on a single-bar-series chart with separated bars. The algorithm by Yokokura and Wantanabe[121] could achieve the claimed accuracy only on hollow or solid bar patterns but not on textured bar patterns. The strict underlying assumption on the chart type restricts the use of their algorithms in recognizing different types of charts which is also the bottleneck in constructing a practical chart recognition system. Chart classification is the solution for the bottleneck and is a key procedure for a sound chart recognition framework. In this chapter, we propose a novel approach for chart classification and chart segmentation by statistical modeling.

Before we explore the topic of statistical modeling for charts, we address the problem of dimension classification by using the information from axes detection.

5.1 Dimension Classification of Charts

After axes detection, the processed chart can be classified into a 2-D chart, a 3-D chart or a chart of other category in terms of the axes information since we have obtained the parametric information about the coordinate system.

Some of the axes in 3-D charts cannot be detected correctly by analyzing the result of axes detection and only two axes are detected for these 3-D charts. If we simply use the number of axes to discriminate the dimension of a chart, the error caused by axes detection will mislead dimension classification. Therefore, we apply the angle constraint to further discriminate the dimensional type of charts. We classify those charts in which three axes are found or two axes are found but their internal angle a as shown in figure 4.3 of chapter 4 is greater than 100° as 3-D charts. Other charts with more than or equal to one axis found are categorized into 2-D charts. Those charts with no axis found are classified as other category which possibly includes pie charts, etc. The result of dimensional classification of charts is illustrated in section 5.5.1.

5.2 Framework of Chart Statistical Modeling

Charts can be classified by extracting graphics primitives such as axes or data markers using image processing plus heuristic searching with the geometric knowledge of the primitives. That means chart classification follows chart segmentation. For comparatively stable graphics primitives like axes, primitive-extraction method for chart classification is advisable since the parameters for they are not too complex and are

tractable. Yet the data markers are versatile and need complex multi-parametric heuristics to segment which makes tuning of the entire system difficult. It is also the reason why currently reported works in [121, 126] can only recognize charts of very strict types. In our work, we propose a new chart classification and segmentation approach based on statistical modeling. Figure 5.1 illustrates the structure of the framework.

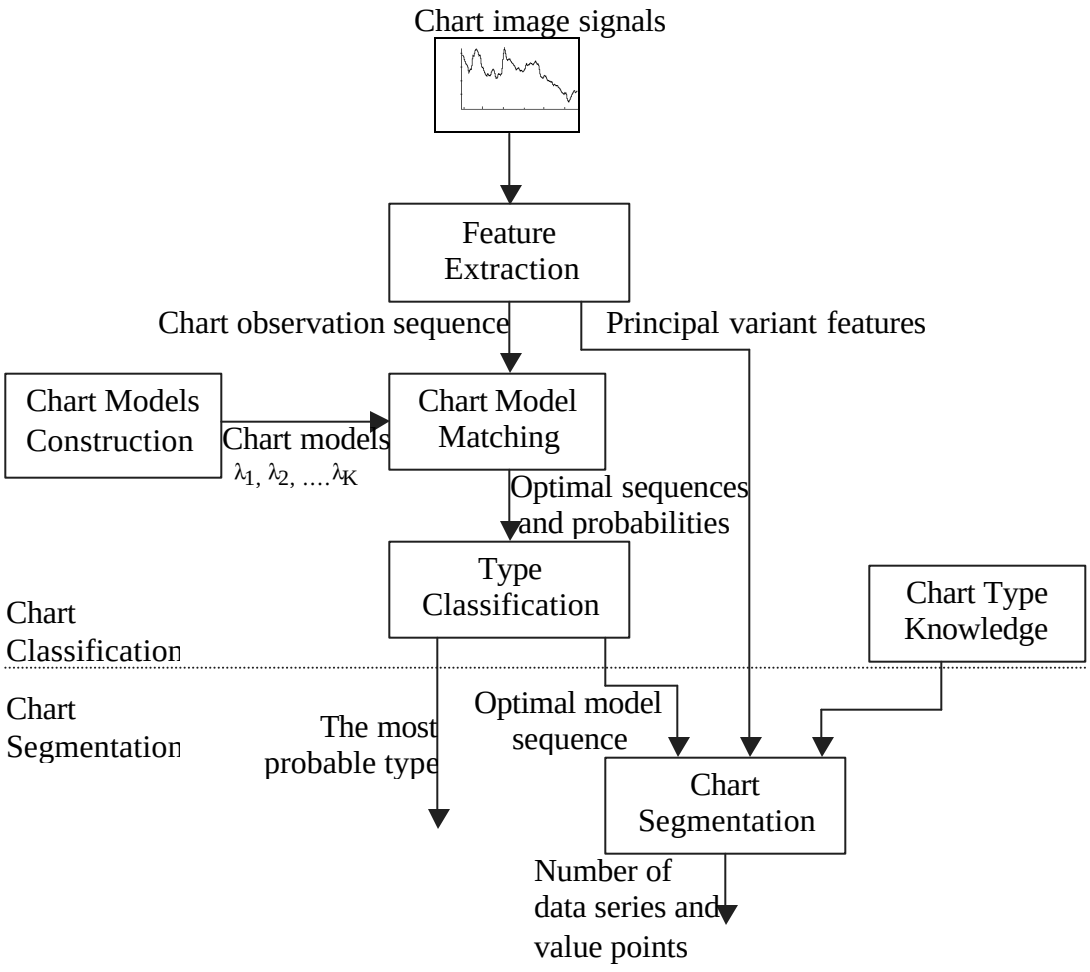


Figure 5.1: Framework of statistical modeling for chart classification and segmentation

Feature Extraction

Unlike the primitive-extraction-based approach, the proposed statistical-model based approach need not explicitly extract data markers in order to classify a chart. An unknown type chart image is converted by a feature extraction processor into a sequence of image observation vectors $O = o_1, o_2, ..., o_T$, where each of these vectors is a compact representation of the short-span image spectrum which contains the necessary image information for recognition. Principal variant feature vectors which record the locations and detailed information for each observation vector are also extracted for later chart segmentation or chart understanding.

Chart Models Construction

The Hidden Markov Model (HMM) is a doubly stochastic process, in which one kind of the variables is hidden, and the other kind of random variables is observable. The HMM has been successfully used in speech processing and recognition and handwritten character recognition in document image analysis domain. Appendix B in the dissertation introduces some basic knowledge about the HMM. The classification and segmentation of graphics symbols in charts stage area have many similarities as compared to handwritten character recognition. They both involve processing of noisy symbols with considerable variations in symbol appearance. The sequence of presentation symbols for charts also share the similarity of time sequence if we view the dimension of X-axis as the dimension of time sequence. We are the first to use the HMM to model the semantic knowledge of graphics symbols in the chart stage area. The repetition of graphics

symbols are also modeled in the HMM by assigning cyclic arcs between state nodes. Each type of chart is modeled by one HMM in which the states may signify the primitive labels of the graphics symbol such as a tick mark, a left side of a bar pattern, or the right side of a bar pattern. The training feature vectors are extracted from the trained images by the feature extraction procedure. The parameters of the models are estimated by a segmental *K*-means algorithm with Baum-Welch re-estimation formulas using maximum likelihood optimization criteria.

Chart Model Matching

The task for chart model matching is to compute the probability of the most probable chart model sequence, $C_k^* = c_{1k}^*, c_{2k}^*, ..., c_{Tk}^*$, given the observed chart image sequence $O = o_1, o_2, ..., o_T$ for each particular chart model l_k, which is denoted $P(C_k^*, O \mid l_k)$, using Viterbi algorithm based on dynamic programming. The optimal state sequence for each model is also obtained by this algorithm.

Type Classification

The optimal chart model l_k in *K* chart models with the highest probability is selected as the chart type in which l_k is calculated using the following equation.

$$l_k = \arg\max_{1 \le k \le K} P(C_k^*, O \mid l_k) \tag{5.1}$$

Chart Segmentation

The optimal state sequence $C_k^* = c_{1k}^*, c_{2k}^*, ..., c_{Tk}^*$ for the detected chart type model l_k is combined with principle variant features from feature extraction and heuristic rules from

chart type knowledge to segment the number of the data series and data markers in a chart. In our chart segmentation problem, we also discuss the primitive-extraction approach with heuristic rules to segment data markers.

The framework of statistic modeling can be divided into two main parts: model-based chart classification and chart segmentation. In the next section, we will address model-based chart classification.

5.3 Model-based Chart Classification

As illustrated in the previous section, model-based chart classification comprises of feature extraction, chart models construction, chart model matching and type classification. Features are selected first before performing the chart type classification. The next section explores the aspect of feature extraction.

5.3.1 Feature Extraction

Extracting appropriate features from charts plays an important role for a robust chart classification and segmentation. Features at the feature points compact the geometrical information. These feature points include the corner points of line segments and rectangles and junction points on the axes. After segmenting the feature points, features are computed to compose an observation vector. These observation vectors are then used in training and testing. We first investigate the problem of feature point detection.

Feature Points Segmentation

To extract the geometrical features, the line which is perpendicular to the X-axis for each pixel along the X-axis is extracted for further processing. Let $PX_{begin}(x_{begin}, y_{begin})$ and $PX_{end}(x_{end}, y_{end})$ denote the two end points of the X-axis. Pixels along the X-axis are denoted as $PX_i(x_i, y_i)$, where $x_{begin} \le x_i \le x_{end}$. We adopt the following slope-intercept parameterization to represent a line l.

$$l : \begin{cases} x = b & vertical \quad lines \\ y = kx + b & non - vertical \quad lines \end{cases} \tag{5.2}$$

A line which is perpendicular to the X-axis at a crossing point $PX_i(x_i, y_i)$, denoted as l_i, is also parallel to the Y-axis whose slope can be computed by the following equation with a known q. Since the parameters of the line l_i are obtained, the pixel values on it can be extracted for further geometric feature extraction.

$$k = -\frac{\cos q}{\sin q} \tag{5.3}$$

The feature points we define in our chart classification are the two pixel points on each visited perpendicular line l_i. One is the crossing point lying on both the X-axis and the extracted perpendicular line l_i, denoted as pixel p_1. The other is the pixel point p_2 lying on the extracted perpendicular line l_i which is farthest from the pixel p_1.

Features Selection

The shapes of these feature points are classified into four groups: T-shape, right-L-shape, left-L-shape and line-end-shape. Start from the center of a feature point and search the boundary of a 10x10 bounding box in four directions: north, south, east and

west as illustrate in figure 5.2. A four-tuple *[N, E, S, W]* represents the pixel extension distribution. Search each direction, if there is pixel extension, then mark the variable in that direction as 1. If the pixel extension distribution for the feature point is equal to *[1,1,0,1]* or [0,1,1,1], we name the shape of the point as T-shape. If the pixel extension distribution for the feature point is equal to *[0,1,1,0]* or [1,1,0,0], we name the shape of the point as left-L-shape which corresponds to the top corner shapes of the left side of a bar. If the pixel extension distribution for the feature point is equal to *[0,0,1,1]* or [1,0,0,1], we name the shape of the point as right-L-shape which corresponds to the top corner shapes of the right side of a bar. If the pixel extension distribution for the feature point is equal to *[0,0,0,1]* or [1,0,0,0], we name the shape of the point as line-end-shape.

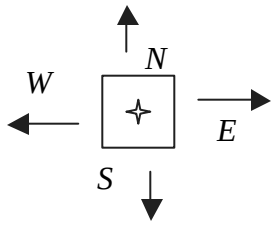


Figure 5.2: Shape analysis for a feature point

The topological and geometrical features are very useful in capturing spatial information of the graphics area. We improve the features in our previous work [129] and present four types of geometrical features: T-shape feature, L-shape feature, line-end-shape feature and transition feature.

- **T-Shape Feature**

T-shape feature describes the information whether or not the pixel p_1 is a T-shape point. Let f_{ts} represent the T-shape feature. It is computed using the following equation.

$$f_{ts} = \begin{cases} 0 & non-T-shape \\ 1 & T-shape \end{cases} \tag{5.4}$$

- **L-Shape Feature**

The attribute of L-shape for the pixel p_2 is extracted as L-shape feature, denoted f_{ls}. It is calculated as follows:

$$f_{ls} = \begin{cases} 0 & non-L-shape \\ 0.5 & right-L-shape \\ 1 & left-L-shape \end{cases} \tag{5.5}$$

- **Line-End-Shape Feature**

Line-end-shape feature contains the information as to whether or not the pixel p_2 is an end-point of a line. Let f_{les} represent the line-end-shape feature. It is computed using the following equation.

$$f_{les} = \begin{cases} 0 & non-line-end-shape \\ 1 & line-end-shape \end{cases} \tag{5.6}$$

- **Transition Feature**

The number of transitions from the foreground to the background for each perpendicular line is also selected as an important feature. The neighboring runs that transition between black and white are merged if the distance between them is within a specific threshold to avoid noise disturbance. The feature f_t which represents the number of the transitions is computed as follows:

$$f_t = \begin{cases} 0 & no\ \ transition \\ 0.25 & one\ \ transition \\ 0.5 & two\ \ transitions \\ 0.75 & three\ \ transitions \\ 1 & four\ \ or\ \ more \end{cases} \tag{5.7}$$

Feature Vectors Generation

The graphical image that we have used to detect axes is also used to generate feature vectors for each chart. A feature vector is extracted at each feature column that is perpendicular to the X-axis. Each chart image is represented by 30 feature vectors in the training process regardless of the actual size of the image. If there exist points with T-shape on the X-axis, the feature vectors at those positions are selected as parts of the training feature vectors for a chart. The feature vectors at the intervals between those positions are also selected with equal spacing to compose the remaining parts of the training feature vectors for that chart. All these vectors are ordered along the X-axis.

The features in the observation sequence are normalized. Another feature vector named the *principal variant feature vector* is used to record the exact information of the feature points in each feature column. The exact information includes the location of the corresponding observation feature vector in the image, the coordinates of the two feature points and transition points which are the reference of value points, the number of transitions and the shape information of each feature point. The principal variant feature vectors will be used in high-level chart understanding.

5.3.2 Chart Model Construction

Chart model construction includes three aspects: chart model selection, topology of chart models and training model parameters.

Chart Model Selection

An HMM has two components, an underlying Markov chain having a finite number of states and a finite set of output probability distributions, one of which is associated with each state.

In an HMM, the observation sequence is $O = o_1, o_2, ..., o_T$ and the chart state sequence from it is $C = c_1, c_2, ..., c_T$. T is the length of the observation sequence. N is the number of states in an HMM. An HMM is normally represented with a compact parameter set $l = (A, B, p)$ [91] where p is the initial state distribution vector, e.g. p_i is the probability of state i at some arbitrary time, t=0. A is the state transition matrix, where A=[a_{ij}], a_{ij} is the probability of transiting to state j given the current state i. B is the output distribution matrix, where B=[b_{jk}], b_{jk} is the probability of observing symbol k given the current state j.

In a discrete HMM, the observation sequence O is required to be finite. The technique of Vector Quantization (VQ) can be applied on the original observation sequence space. An observation sequence is quantized by substituting it to the closest element from a finite codebook of vectors. For most applications, the observations are continuous signals. VQ of these continuous signals can degrade performance significantly. Moreover, the codebooks generated by the quantization process are constructed using

training data from all classes. When a new class of training data is added, we need to reconstruct the codebook and retrain all the HMM. But if the observation densities are continuous, we do not need to train the system from the beginning since there is no codebook to be constructed, we only need to train the newly added class. Thus in our work, we adopt continuous HMM with *M*-component Gaussian mixture to model charts.

In a continuous HMM, for each state j it is necessary to compute the probability $b_j(o)$ for each observation vector o in a continuous space. The probability of each observation vector for each state j with one-dimensional Gaussian of mean m_j and covariance U_j is computed using equation (5.8).

$$b_j(o) = \prod_{k=1}^{K} \frac{1}{\sqrt{2pU_{jk}}} \exp\left(-\frac{1}{2}\left(\frac{o_k - m_{jk}}{U_{jk}} \right)^2 \right) \tag{5.8}$$

where o_k is the k^{th} feature of o, m_{jk} is the mean of the distribution of the k^{th} feature of state j and U_{jk} is the variance of the k^{th} feature of state j.

In our current work, we research into two main kinds of charts: bar charts and line charts. The fill-in patterns of the bar charts can be of a large variety since we arbitrarily collect the data set. The one-dimensional Gaussian distribution is not sufficient to model the variation of the fill-in patterns of bar charts. Thus we adopt a more complicated distribution function, i.e., *M*-component Gaussian mixtures. An *M*-component Gaussian mixture is a linear combination of *M* Gaussian probability density function calculated by the equation (5.9).

$$b_j(o) = \sum_{m=1}^{M} W_m b_j^m(o) = \sum_{m=1}^{M} W_m \left\{ \left\{ \prod_{k=1}^{K} \frac{1}{\sqrt{2pU_{jk}^m}} \exp\left(-\frac{1}{2}\left(\frac{o_k^m - m_{jk}^m}{U_{jk}^m} \right)^2 \right) \right\} \right\} \tag{5.9}$$

where W_m is the weight of mixture m.

Topology of Chart Models

The left-to-right hidden Markov models are a commonly used topology in speech recognition and handwritten character recognition because this topology is more suitable to model constrained temporal order. In our processing, the chart sequence shows a repetition characteristic. The ergodic topology can represent such characteristic better than the left-to-right model. In such topology, it is possible to reach any state from any other state.

In our work, we investigate two kinds of charts, bar charts and line charts, since they are the most commonly used charts conveying both discrete and continuous trend information.

Although the bar chart images we collected show diversity in the fill-in texture bar patterns and difference in the number of bar series, they can be generally grouped into two categories, separated bar charts and contiguous bar charts, according to the inter-bar spacing. In a separated bar chart, the value of the inter-bar spacing is positive which means each bar in it is not connected with each other. In a contiguous bar chart, there are bars connecting with each other and the value of the inter-bars spacing between them is not positive. Almost all the multiple-bar-series charts in our data set are contiguous bar charts. People are used to represent a group of data comparison by a group of connected bars and separate each group with a positive inter-group spacing. Most of the single-bar-series charts in our collection are separated charts. But the contiguous single-bar-series charts are also commonly used. The difference between a contiguous single-bar-series chart and a contiguous multiple-bar-series chart is that there is no inter-group spacing in

a contiguous single-bar-series chart. By combining with the semantic information of bar charts, we construct two bar chart models based on HMM as shown in subfigure (a) and (b) in figure 5.3. Figure 5.3 (a) is the separated bar chart model. And figure 5.3 (b) is the contiguous bar chart model. The number of the hidden states of the models is decided by combining our experience on adjusting the HMM and the semantics in the chart stage area. The states of the bar models have corresponding semantic meaning. State 1 signifies the Y-axis. State 2 represents the non-bar area. State 3 represents the left side of a bar. State 4 signifies the inner bar area or the texture area of a bar. State 5 represents the right side of a bar.

We construct two line models for line charts: the single-line-series chart model and the multiple-line-series chart model. Subfigure (c) and (d) in figure 5.3 show the topologies of these two models. The semantic meaning for each state in the line models is not as clear as that in the bar models. State 1 for both line models represents the Y-axis. In the single-line-series model, state 2 may represent the line area and state 3 the non-line area. In the multiple-line-series model, state 2 may represent the line area, state 3 the non-line area and state 4 other area.

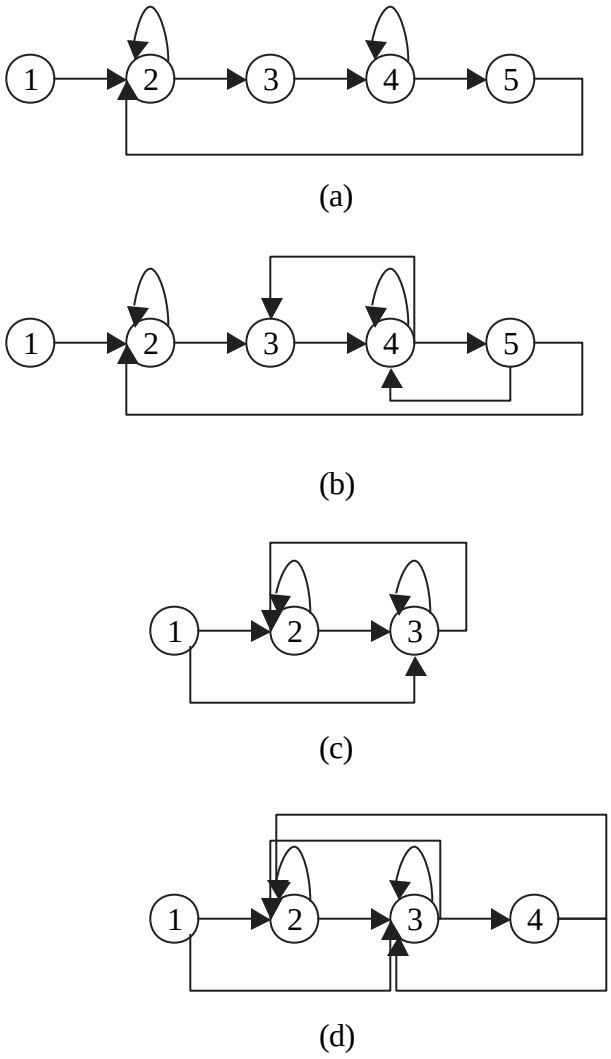


Figure 5.3: Topologies of HMM-based chart models. (a) Topology of the separated bar chart model. (b) Topology of contiguous bar chart model. (c) Topology of single-line-series model. (d) Topology of multiple-line-series model.

Training Model Parameters

In order to train the parameters for each model, a segmental *K*-means algorithm with Baum-Welch re-estimation formulas is used to re-estimate parameters of the models. We assume diagonal covariance matrices for the Gaussian mixtures [80].

We first define two variables: the forward variable $a_t(i)$ and the backward variable $b_t(i)$ using equations (5.10) and (5.11) and compute them using equations (5.12) to (5.15) [91].

$$a_t(i) = P(o_1, o_2, ...o_t, state\ i\ at\ time\ t\ /\ l) \tag{5.10}$$

$$b_t(i) = P(o_{t+1,} o_{t+2}, ...o_T\ /\ state\ i\ at\ time\ t\ ,\ l) \tag{5.11}$$

$$a_1(\mathrm{i}) = p_i b_i(\mathrm{o}_1) \tag{5.12}$$

$$a_{t+1}(\mathrm{j}) = \sum_{\mathrm{i}=1}^{\mathrm{N}} a_t(\mathrm{i}) \mathrm{a}_{\mathrm{ij}} \mathrm{b}_{\mathrm{j}}(\mathrm{o}_{t+1}) \tag{5.13}$$

$$b_T(\mathrm{i}) = 1 \tag{5.14}$$

$$b_t(\mathrm{i}) = \sum_{\mathrm{j}=1}^{\mathrm{N}} \mathrm{a}_{\mathrm{ij}} \mathrm{b}_{\mathrm{j}}(\mathrm{o}_{t+1}) b_{t+1}(\mathrm{j}) \tag{5.15}$$

The probability of an observation sequence for a given model l is computed by equation (5.16). The parameters of re-estimation model $\overline{l}$ such as the transition probability $\overline{\mathrm{a}_{\mathrm{ij}}}$, the probability distribution, $\overline{\mathrm{b}_{\mathrm{jk}}}$, and the initial state probability $\overline{p_{\mathrm{i}}}$ are calculated using the Baum-Welch re-estimation formulas of equations (5.17), (5.18) and (5.19) respectively.

$$P(O \mid l) = \sum_{i=1}^{N} a_T(i) \tag{5.16}$$

$$\overline{a_{ij}} = \frac{\sum_{t=1}^{T-1} a_t(i) a_{ij} b_j(o_{t+1}) b_{t+1}(j)}{\sum_{t=1}^{T-1} a_t(j) b_t(j)} \tag{5.17}$$

$$\overline{b_{jk}} = \frac{\sum_{t \ni o_t = k} a_t(j) b_t(j)}{\sum_{t=1}^{T} a_t(j) b_t(j)} \tag{5.18}$$

$$\overline{p_i} = \frac{1}{P} a_1(i) b_1(i) \qquad \text{where } P = P(O \mid l) \tag{5.19}$$

Figure 5.4 illustrates the main procedures of the segmental *K*-means training algorithm.

1. Choose initial values for model parameters.
2. Refine the initial parameters using Baum-Welch re-estimation formulas.
3. Segment training sequence into *N* states.
4. Cluster the training vectors for each state into *M* clusters and calculate the distribution of the Gaussian mixtures.
5. Update the model parameters from the newly segmented states and clusters.
6. Refine model parameters using Baum-Welch re-estimation formulas.
7. If the new model is not better than the old model, stop re-estimation, otherwise go to step 3.

Figure 5.4: Segmental *K*-means training algorithm for chart models

We select the initial parameters empirically by considering common distribution knowledge of graphics symbols in chart stage area. To segment the sequence into N states, we use an optimal state sequence finding algorithm, Viterbi algorithm, which is

also used in type classification with chart model matching. The Viterbi algorithm will be illustrated in detail in the next section. The number of hidden states for chart models is known and fixed. The Gaussian mixture M for state 4 in the two bar models whose semantics denotation is the fill-in texture of bar patterns varies from $M=1$ to $M=3$. The Gaussian mixtures for other states are set to $M=1$.

5.3.3 Type Classification by Chart Model Matching

A testing chart is processed to extract testing feature vectors as the observation sequence. We have trained four chart models for four different chart classes l_k, where $1 \le k \le 4$. The typical solution for type classification use the probability of the observation vectors given a known model λ_k, $P(O|l_k)$, to do classification.

In our work, the optimal state sequence is also used for chart segmentation. We name the probability of the most probable chart model sequence, $C_k^* = c_{1k}^*, c_{2k}^*, ..., c_{Tk}^*$, given the observed chart image sequence for each particular chart model l_k, which is denoted $P(C_k^*, O \mid l_k)$, as the chart model matching score. Type classification is to classify a chart into the model type that has the highest chart model matching score as in equation (5.1). The Viterbi algorithm [91] is applied in chart type classification using a form of dynamic programming and a trellis structure that efficiently implements the computation. Let $y_t(i)$ denote the survivor terminating in state i_t and $d_t(i)$ denote the survivor score in i_t, where $1 \le t \le T, 1 \le i \le N$. The Viterbi algorithm is shown in figure 5.5.

The Viterbi algorithm consists of the following fours steps.

Step 1—Initialization: for $1 \le i \le N$

$$d_1(i) = p_i b_i(o_1) \tag{5.20}$$

$$y_1(i) = 0 \tag{5.21}$$

Step 2—Recursion: for $2 \le t \le T, 2 \le i \le N$

$$d_t(i) = \max_{1 \le i \le N}[d_{t-1}(i)a_{ij}]b_i(o_t) \tag{5.22}$$

$$y_t(i) = \arg\max_{1 \le i \le N}[d_{t-1}(i)a_{ij}] \tag{5.23}$$

Step 3—Termination:

$$P^* = \max_{1 \le i \le N}[d_T(i)] \tag{5.24}$$

$$c_T^* = \arg\max_{1 \le i \le N}[d_T(i)] \tag{5.25}$$

Step 4—Path backtracking:

For *t=T-1, T-2,…, 1*

$$c_t^* = y_{t+1}(c_{t+1}^*) \tag{5.26}$$

Figure 5.5: Viterbi algorithm for Hidden Markov Model

5.4 Chart Segmentation

After a chart is classified into its corresponding type, domain semantic knowledge for this chart type can be used to segment the data markers and detect the value points for further chart interpretation. The issue of detection of the number of data series in a chart is also a problem in chart segmentation so as to extract correct positions of value points.

Low-level heuristic searching method is the main approach to perform chart segmentation. In our work, we also propose a new chart segmentation approach by clustering and searching the optimal state path from chart classification. We first explore the traditional low-level heuristic searching approach.

5.4.1 Chart Segmentation by Low-Level Heuristic Search

We apply this approach to segment bar patterns in a separated bar chart and detect the number of data-series in a multiple-line-series line chart.

Bar pattern Segmentation

The general idea to segment bar patterns is to detect the line segments from pixel processing and then cluster the line segments to search the rectangles as bar pattern candidates. Other domain knowledge about bar pattern such as spacing between bars can be used to verify the bar pattern heuristically.

Our earlier works in [127, 128] endeavored in recognizing bar patterns starting from the pixel level. We had not addressed the problems of plot area detection and axes detection during that time. The underlying assumptions for the earlier works are that what we process is a 2-D bar chart in which the bars are separated and distributed along

the axes with equal space. Two bar pattern detection algorithms were proposed for reconstructing the bar patterns. The first one is called hypothesis-testing bar pattern searching algorithm which searches the Hough space for bar patterns. It first applies the standard Hough transform on the chart image to obtain the accumulator map. The accumulated map is enhanced by a butterfly filter proposed by Leavers [66]. Hypothesis on the Y-axis and X-axis candidates is generated according to the number of the peak points in each thresholded Hough division. Line segments in the image space are reconstructed to verify the axes. Consecutive parallel line segments in the Y-axis area are grouped as the bar candidates and top line segments of the bar patterns are reconstructed to verify these bar patterns. The other approach to extract bar patterns is called Modified Probabilistic Hough Transform algorithm (MPHT) [126]. Unlike the first approach in which parallel line segments are detected by first voting all the foreground features into the accumulator map and then verify the line peaks in the image space, MPHT votes part of the pixels to get a line candidate and its parallel line candidates, and then extract the pixels by verifying the line candidate and possible parallel line candidates in the image space until no foreground features are left. After parallel lines are detected, they are grouped into bar patterns sequentially using simple heuristics. The advantage of MPHT is to cut the cost of voting by probabilistically selecting and minimizing the voting features using dynamical line verification in the image space. Detailed illustration of these algorithms and experiments can be found in [127, 128].

As in our low-level processing, we have already applied Hough transform in axes detection. We have obtained the information of lines segments during the procedure of axes detection which can be used in extracting bar patterns. Thus no other independent Hough transform is needed. Therefore the abovementioned two Hough-based bar pattern

detection approaches are not used in current work. The pairs of neighboring parallel line segments can be looked upon as both sides of bars. Yet the accuracy of bar pattern detection based on this simple scheme will degrade a lot if one line segment in its corresponding pair is missing. Thus the attributes like the length of the line segments are useful to stabilize the bar pattern reconstruction. In the previous axes detection process, the line segment is calculated to see whether it is related with its neighboring parallel line segments. A program using lines segments and their attributes detected in the low level processing to extract bar patterns in single-bar-series chart is as follows:

1. Group those related neighboring line segments paralleling to the Y-axis as side pairs of bars and mark them as visited.
2. Starting from an unvisited line segments closest to a detected bar, find its unmarked neighboring line segments. If not found, mark it as visited and go to step 3. If found, group it with its neighbor as the sides of a bar and mark both line segments as visited.
3. Repeat step 3 until all the line segments are visited. The grouped bars are the detected bar patterns.

Figure 5.6: Algorithm of bar pattern segmentation by primitive extraction

Detecting the Number of Line-Series

A simple program is developed to estimate the number of data series of multiple-line-series charts. First, for each column along the X-axis, count the number of the runs which

denote the transition between foreground and background. Second, histogram the run numbers. The peak of the histogram is then selected as the number of the line series.

5.4.2 Chart Segmentation by Optimal Path Clustering

After chart classification, an optimal state sequence of the categorized chart is obtained. The hidden states of the chart models are closely related to the semantics of the chart. We propose the idea of segmenting a chart by searching and clustering the optimal state sequence. We apply the optimal path clustering approach to solve two problems. The first is to segment bar pattern in separated bar charts. The other is to detect the number of data series in a contiguous bar chart.

Bar Pattern Segmentation

The same consecutive states in the optimal state sequence are first clustered into one state. The middle position of the state in the clustered states is assigned as the position of the new state.

The algorithm of bar pattern segmentation by optimal path clustering is shown in figure 5.7.

1. Cluster the same consecutive states into one state.
2. Set the position of the first found state 3 in the unchecked sequence as a start of a bar.
3. Check the next state. If no unchecked state is left, stop the program.
4. If the next state is state 3, set the position of it as a start of a bar, go to step 3.
5. If the next state is state 5, set the position of it as the end of the bar.
6. Get the positional information for the start and end of a bar from the principal variant feature vector to reconstruct the bar. Go to step 2.

Figure 5.7: Algorithm of bar pattern segmentation by optimal path clustering

Detecting the Number of Bar-Series

Figure 5.8 shows the procedure of detecting the number of bar-series by optimal path clustering.

1. Cluster the same consecutive states into one state. Delete the first state if it is state 1.
2. Find the positions of state 2 in the sequence.
3. Segment the sequence into bar groups at the positions of state 2.
4. For each bar group, compute the number of state 4.
5. The maximum of the number is selected as the number of the bar series

Figure 5.8: Detecting the number of bar-series by optimal path clustering

5.5 Experiments and Analysis

5.5.1 Experiments on Chart Classification

Chart classification involves two problems: dimension classification and type classification. In the experiments of chart classification, the precision means the proportion of the correctly classified charts of a category to all classified charts of that category. The recall measures the probability of the correctly classified charts of a category to all existing charts of that category.

Results of Dimension Classification

Among the total 537 testing images, 493 images are two-dimensional charts while the other 44 images are three-dimensional charts. After classification, 487 images are recognized as two-dimensional charts and 45 images are classified as three-dimensional charts. The remaining five charts are classified into other category. These five data charts are misclassified into other category charts due to the incomplete plot area detected**.** The axes are not included in the incomplete plot area because of the serious brokenness of the axes. One two-dimensional chart is misclassified as a three-dimensional chart because three axes are detected in the plot area. The following table evaluates the performance of chart classification on dimension of the coordinate system.

Performance / Categories	Total	Detected	Correct	Precision (%)	Recall (%)
2-D charts	493	487	487	100%	98.78%
3-D charts	44	45	44	97.78%	100%

Table 5.1: Performance evaluation for dimension classification

From the result of table 5.1, we can see that using the axes information from axes detection to classify charts into 2-D charts or 3-D charts is robust. The dimension classification can exclude the 3-D charts so that we can process and interpret only 2-D charts.

Results of Type Classification

We use about 75% of each category of charts to train the parameters of each model. Since the charts used for training a particular model are not used to train other models, we mix all the trained and untrained images as the test set for type classification. The output of type classification is the number of its recognized type. Table 5.2 shows the result of type classification on all the images.

Chart Type	Total	Classified	Correct	Precision	Recall
Separated bar chart	87	100	83	83%	95.4%
Contiguous bar chart	98	85	81	95.29%	82.65%
Single-line-series chart	55	71	52	73.23%	94.5%
Multiple-line-series chart	142	126	123	97.62%	86.62%

Table 5.2: Performance evaluation for type classification

The structures of the bar models and the line models differ a lot. Thus the chart can be correctly classified into two big categories: bar charts or line charts. But in each category, due to the similar topology for each type, wrong classification occurs easily. Table 5.2 shows that the recalls of both separated bar charts and single-line-series line charts are higher than those of their corresponding counterparts. While the precisions of them are

lower than those of their corresponding counterparts. In some contiguous bar charts with two-data-series, some data elements for a series are zero and no bar patterns are used to represent zero. Thus some two-bar-series contiguous bar charts are misclassified as separated bar charts. Some separated bar charts where the inter-bar spacing is small are misclassified as contiguous bar charts. More charts are classified into single-line-series line charts. The main reason for it is that line segments of the dashed lines are missing in the graphic image used for feature extraction. The scarcity of training data in single-line-series may also be the reason of low precision of the single-line-series charts. Nevertheless, the result of type classification is still encouraging.

5.5.2 Experiments on Chart Segmentation

We address two problems in chart segmentation: detecting the number of data series and bar pattern segmentation.

Detecting the Number of Data Series

In our collected data set of multiple-data-series charts, most of them are two-data-series and three-data-series charts. Charts with higher order data series are scarce. Thus in the following table 5.3, we report the results of detecting the number of the data series of the following charts: two-bar-series charts, two-line-series charts and three-line-series charts. The number of the data series for multiple-bar-series charts are detected by optimal path finding and the number of the data series for multiple-line-series charts are detected by a simple low level processing approach. The precision measures the proportion of the correctly detected number of data series for a category to all detected number of data

series for that category. The recall measures the proportion of the correctly detected number of data series for a category to the ground truth for that category.

Data markers	Total	Detected	Correct	Precision	Recall
Two-bar-series	78	74	63	85.1%	80.77%
Two-line-series	58	50	39	78%	67.24%
Three-line-series	34	36	28	77.8%	82.36%

Table 5.3: Results of detecting the number of data series of multiple-data-series charts

The fact that some bar charts with more than two bar series are misclassified as two-bar-series charts pulls the precision rate down. While misclassification of a real two-bar-series chart into a separated bar chart makes the recall rate lower. We check the real graphical images that fails in the detection of the bar series and find that missing of bars representing one data series may cause the behavior of the optimal state path uncertain in its semantic meaning. In detecting the number of the line series, the problem of missing of dashed line segments that lessen the performance of type classification is also the main reason to deteriorate the performance of the data series detection. One possible solution for it may be by adding a special dashed line segments analysis and restore them in the graphical image for feature extraction.

Bar Pattern Segmentation

The procedure to segment bar patterns is to find the value point of the data markers. We have proposed two approaches to segment bar patters on separated bar charts: low level heuristic approach and optimal path finding. The outputs of bar pattern segmentation are quadrilaterals that represent the detected bar patterns. The points on the top line of the

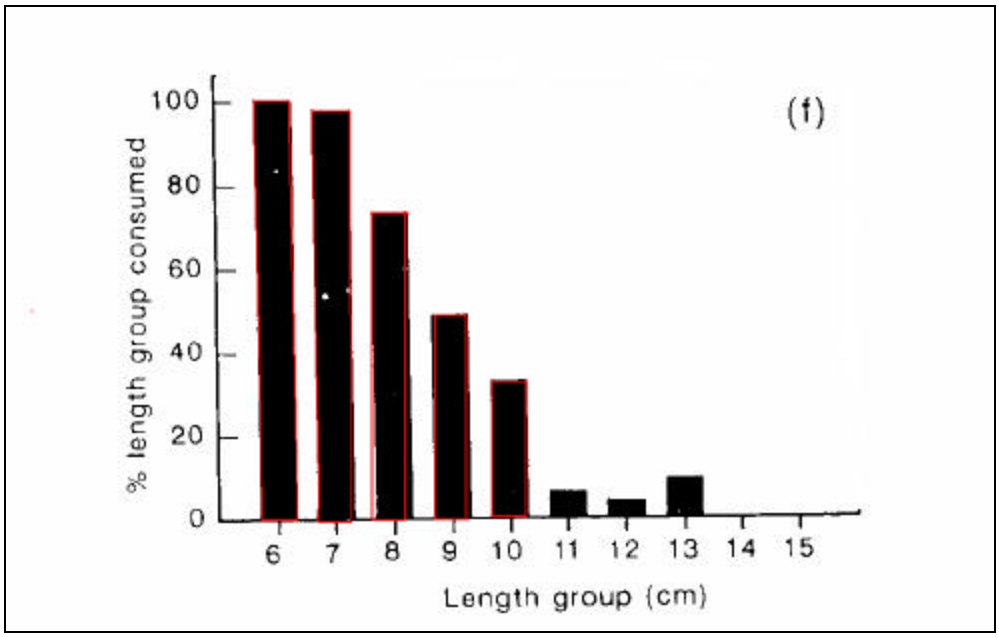
bar pattern are the corresponding value points for that bar pattern. Figure 5.9 illustrates an example of bar pattern segmentation with two approaches on a simple bar chart. Some short bar patterns are not detected in the approach of low-level heuristic search. In this example, all the bar patterns are segmented correctly by the optimal path finding approach. We selected 50 separated bar charts for testing from which we obtained a total of 216 obvious bar patterns. The precision is the proportion of correct bar patterns to all the detected bar patterns. The recall means the proportion of correct bar patterns to all existing bar patterns. The following table 5.4 shows the results of two approaches in segmenting bar patterns of separated bar charts.

Bars / Methods	Detected	Correct	Precision	Recall
Low-level heuristic search	180	156	86.67%	72.22%
Optimal-path-finding	187	142	75.94%	65.74%

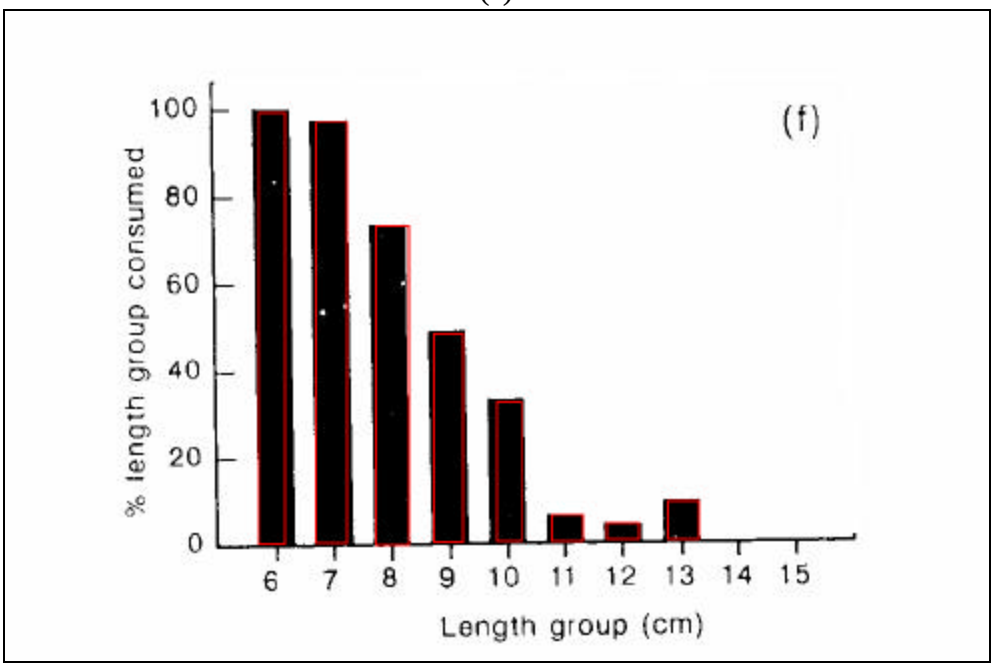
Table 5.4: Results of detecting bar patterns for separated bar charts

Table 5.4 shows the overall performance of the low-level heuristic search approach is better than that of the optimal path finding approach. Although the states in the separated bar model may correspond to some particular semantic meaning, the uncertainty of the hidden states may be the reason of incorrect bar pattern segmentation. Most of the short bar patterns cannot be detected by the low-level heuristic search approach. The reason for it is due to the difficulty in adjustment for the threshold of the length of line segments. If we lower the threshold to allow more short line segments, more spurious line segments detected will also deteriorate the performance of the bar pattern segmentation. On the other hand, the optimal path finding approach does not meet such a dilemma. Thus in the

future work, it is promising to obtain higher performance by combining both approaches appropriately.



(a)



(b)

Figure 5.9. Results of bar pattern segmentation approaches on a separated bar chart. (a). Result of the low-level heuristic search approach. (b). Result of optimal path finding approach. Note: the red quadrilaterals represent the segmented bar patterns.

5.6 Summary

A new framework for chart classification and segmentation based on statistical modeling is proposed. Four chart models including separated bar model, contiguous bar model, single-line-series line model and multiple-line-series line model are constructed and trained using a segmental *K*-means algorithm to model the semantics of chart stage area. Charts are classified by choosing the chart model with the largest posteriori probability. The best state path for that model is obtained by applying Viterbi algorithm. Two kinds of classifications, dimension classification and type classification, are addressed. A new approach for chart segmentation using optimal path finding is proposed. Two chart segmentation problems are addressed, including detecting the number of data series and bar pattern segmentation.

The contributions of this chapter include:

1. We propose a new framework for chart classification and segmentation based on statistical modeling.
2. We propose a model-based chart classification approach. This includes feature extraction with feature point segmentation and analysis, construction and training of HMM-based chart models, type classification by chart model matching.
3. We propose a new approach for chart segmentation by optimal path clustering and finding.

Chapter 6

Text Primitive Analysis and Chart Interpretation

The characters inside a chart are distributed sparsely comparing with mostly-text documents. Nevertheless, they are organized by a regular spatial relationship into different levels of groups with highly meaningful logical meaning that is called *text primitives* in our work. Thus words are one type of text primitives that has lower level logical meaning, while the title of the chart is another type of text primitives with a higher-level logical meaning. *Text primitive analysis* is to discover the formatting of the text elements in a document image so as to derive the meaning associated with the positional and functional blocks in which the text is located. Text primitive analysis is the main procedure of *page layout analysis* in mostly-text documents. Besides labeling the text areas and associating logical meaning such as paragraphs with them, page layout analysis also labels the areas of halftones and line drawings in a document image. In mostly-graphics document images such as engineering drawings, diagrams and charts, etc, text primitive analysis can use orientation results from graphics primitive recognition

and accordingly provide helpful feedbacks for graphics primitive recognition, and by being correlated with the result from graphics primitive recognition, provide semantic understanding of a chart or a diagram, i.e., *chart interpretation*.

The X-Y tree structure proposed by Nagy and Seth [83] is widely used to represent the hierarchical structure of a page layout. The X-Y tree is a nested decomposition that hierarchically subdivides a document page by recursive X-Y (X-horizontal and Y-vertical) cuts. At each level of the X-Y tree, subdivision in only one direction, either horizontal or vertical, is considered. Thus a well-ordered examination sequence starting from the root node of an X-Y tree is established. Traditional X-Y tree or its variations take the whole document image as the root node for starting decomposition. Such decomposition can represent almost all the Manhattan structure of document pages. But the classical X-Y tree decomposition can hardly be applied on non-Manhattan structure of document pages.

Texts in charts or diagrams are sparsely distributed but structurally arranged. For example, the tick names for the Y-axes are arranged in the same column and aligned in one of the three types: centered, flush left or flush right. For most of the 2-D charts without serious skew, the texts are in a Manhattan format. Hence the X-Y tree is a good choice to represent the hierarchical structure of the texts inside a chart. But the texts in many skewed 2-D charts and 3-D charts cannot be represented by the classical X-Y tree structure since they are not in a Manhattan format. In this dissertation, we propose a zoned directional X-Y tree structure which uses known directions from graphics symbol recognition to direct a well-ordered sequential decomposition. The zoned directional X-Y tree we proposed can be viewed as a generalized version of the classical X-Y tree. We describe it in the next section.

6.1 Zoned Directional X-Y Tree Structure

Figure 6.1 illustrates the structure of zoned directional X-Y tree which hierarchically represents the structural relationship of text elements in a mostly-graphics document. The root node is the entire document image with only text elements. There are two aspects related with the proposed tree structure: *zoned* and *directional*.

Normally, traditional recursive X-Y cut segmentation is applied on the whole area of mostly-text documents such as newspapers or book chapters but will fail if applied directly on mostly-graphics document images. For mostly-graphics documents such as maps, engineer drawings, or charts, text elements are distributed with a close relation with its corresponding graphical zone. For instance, text elements are distributed regularly in different areas of a chart, such as in the stage area or in the axes major area. By text/graphic separation, the relation between the texts and their corresponding graphical zone is also removed. Therefore the X-Y trees generated from the document image that contains only text elements cannot reflect the correct structural relationship of texts for the mostly-graphic documents. In our processing, the entire document image is divided into different independent areas by graphical processing. Every segmented area is then inserted under the root node as a sub-root node which is also called the zone node. These zone nodes contain the orientation information from the graphical processing. Further directional X-Y decomposition starts from every sub-root node.

Suppose a document is divided into N zones each having a zone angle q_i as shown in figure 6.1. Unlike classical recursive X-Y cuts where X represents horizontal direction and Y represents vertical direction, the X direction q_{xi} and the Y direction q_{yi} in a directional X-Y tree whose zone angle is q_i are calculated as follows:

$$q_{xi} = \begin{cases} q_i & 0^\circ \le q_i < 90^\circ \\ (q_i + 90^\circ) \bmod 180^\circ & 90^\circ \le q_i < 180^\circ \end{cases} \tag{6.1}$$

$$q_{yi} = \begin{cases} q_i & 90^\circ \le q_i < 180^\circ \\ q_i + 90^\circ & 0^\circ \le q_i < 90^\circ \end{cases} \tag{6.2}$$

The classical X-Y tree is a special case of the directional X-Y tree where $q_i = 0^\circ$. Each node in the direction X-Y represents a rectangle lying in its X or Y direction. The children of a node are obtained by subdividing the directional rectangle of the parent node either in q_{xi} direction or in q_{yi} direction, with the two directional cuts being alternately applied on successive levels of the tree. The leaves are the nodes of inseparable directional rectangles.

The directional X-Y tree keeps the main properties of the original X-Y tree. At each level, only one directional cutting must be considered and the directional cutting in the successive level alternates strictly. Only directional rectangles are generated which allows identical processing at every level. Since the subdivided rectangles are directional, the format of the document layout need not be in a Manhattan format. Thus the directional X-Y tree structure can be viewed as a generalized version of classical X-Y tree.

We use a zoned directional X-Y tree to hierarchically represent the spatial relationship of the text primitives in charts. It can also be applied to other categories of diagrams to represent the textual relationship in graphical documents. For example, in a map, the street names or labels along a street can be represented into a street label X-Y tree if the direction of the street is known.

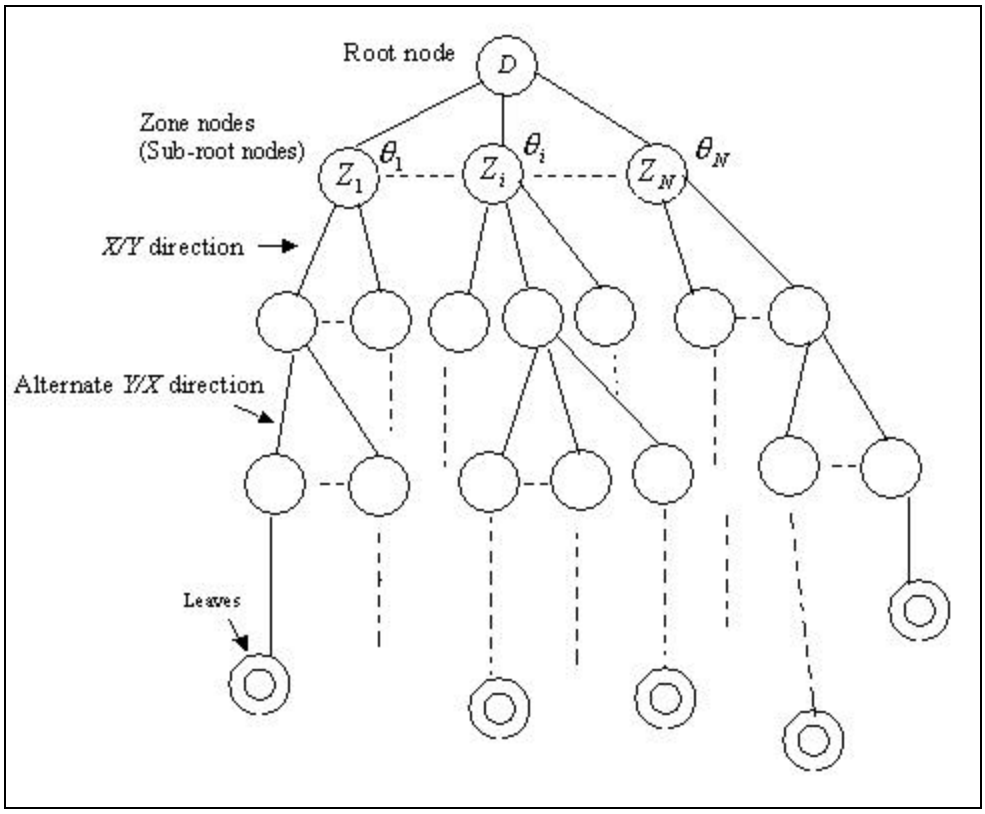


Figure 6.1: Structure overview of a zoned directional X-Y tree. *D*: the root node which represents the entire document with only text elements. $Z_1 ... Z_i ... Z_N$: N sub-root zone nodes each with an angle q_i, where $1 \le i \le N$.

6.2 Zoned Directional X-Y Tree Generation

We use the projection profile of bounding boxes instead of traditional pixel projection to generate the zoned directional X-Y tree. Ha *et al.* [42] also proposed the similar idea of using bounding box projection profile. We discuss the differences of the two approaches in the section 6.2.2. If the direction of the zone is not horizontal or vertical, directional transforms are first applied on all the bounding boxes inside the zone.

6.2.1 Directional Transform for the Bounding Boxes

The directional transform we propose is to transform the bounding boxes lying in a non-orthogonal orientation in the image space into the bounding boxes lying in an orthogonal orientation in a space that we name as *r-space* as illustrated in the figure 6.2.

Suppose we are given a bounding box b in a directional zone whose X direction and Y direction are q_x and q_y respectively. The bounding box b encloses a region R in the image space as shown in figure 6.2 (a), where

$$R = \{(x, y) \mid x_{\min} \le x \le x_{\max} \text{ and } \quad y_{\min} \le y \le y_{\max} \} \tag{6.3}$$

Its directional transformed bounding box b' encloses a region R' in the r-space as shown in figure 6.2 (b), where

$$R' = \{(rx, ry) \mid rx_{\min} \le rx \le rx_{\max} \text{ and } \quad ry_{\min} \le ry \le ry_{\max} \} \tag{6.4}$$

We use the polar form to represent a straight line. The r-space is named because it actually represents the r distribution in the X and Y directions.

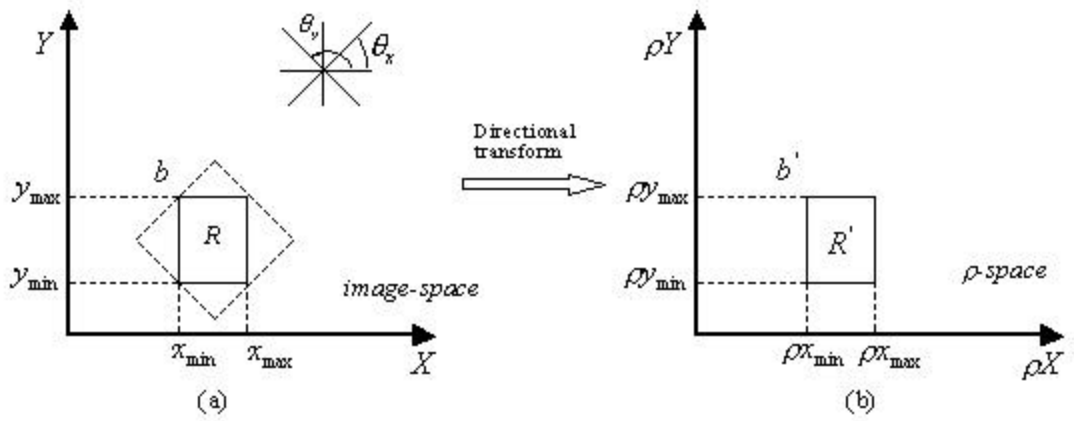


Figure 6.2: Illustration of directional transform on a bounding box. (a): a bounding box b in a zone whose X direction and Y direction are q_x and q_y respectively. The dashed rectangle enclosing the bounding box b is the transformed bounding box shown in the image space. (b): a transformed bounding box b' in the *r-space.*

Four straight lines on the transformed bounding box shown in the figure 6.2 (a) as the dashed rectangle can be calculated by the following four equations:

$$ry_1 = x_{\min} \cos q_y + y_{\min} \sin q_y \tag{6.5}$$

$$ry_2 = x_{\max} \cos q_y + y_{\max} \sin q_y \tag{6.6}$$

$$rx_1 = x_{\min} \cos q_x + y_{\max} \sin q_x \tag{6.7}$$

$$rx_2 = x_{\max} \cos q_x + y_{\min} \sin q_x \tag{6.8}$$

Thus the transformed bounding box b' in the *r-space* can be computed using the following equations (6.9) to (6.12).

$$rx_{\min} = MIN\{rx_1, rx_2\} \tag{6.9}$$

$$rx_{\max} = MAX\{rx_1, rx_2\} \tag{6.10}$$

$$ry_{\min} = MIN\{ry_1, ry_2\} \tag{6.11}$$

$$ry_{\max} = MAX\{ry_1, ry_2\} \tag{6.12}$$

The algorithm of directional transform of bounding boxes is shown in the following figure 6.3.

1. For a directional zone, calculate q_x and q_y by using equation (6.1) and (6.2).
2. List all the bounding boxes in this zone.
3. For each bounding box, apply directional transform on it by using the eight equations from (6.5) to (6.12).
4. Repeat step 3 until all the bounding boxes are transformed.

Figure 6.3: Algorithm of directional transform for the bounding boxes.

6.2.2 Recursive X-Y Cut by the Bounding Boxes

In a zoned directional X-Y tree, the root node is not the starting point to do decomposition. Decomposition is performed in every divided zone starting from every sub-root node. Recursive X-Y cut segmentation algorithm is usually used to segment documents with Manhattan layout in the area of document layout analysis. Our generation algorithm for the directional X-Y trees starting from the zone nodes is also based on recursive X-Y cut segmentation method, but has major differences from the traditional recursive X-Y cut segmentation algorithm.

Unlike traditional recursive X-Y cut algorithm [83] using the projection profile of image pixels, our recursive X-Y cut segmentation method uses the projection profile of the bounding boxes. Ha *et al.* [42] also proposed the similar idea of using bounding box

projection profile in the recursive X-Y cut algorithm. One main difference between our algorithm and the algorithm of Ha *et al.* is that the bounding boxes they adopted are just the bounding boxes in the image space. In our work, the bounding boxes for projecting can be the bounding boxes of the character-like connected components if the orientation of the zone is horizontal or vertical, or the directional transformed ones in the *r-space* if the orientation of the zone is neither horizontal nor vertical. Another main difference is the different definition of the vertical and horizontal projection functions for the bounding boxes.

Suppose we are given a set of bounding boxes $B = \{b_1, b_2, \cdots b_N\}$ where each b_i encloses the region

$$R_{b_i} = \{(x,y) \mid x_{b_i,\min} \le x \le x_{b_i,\max} \; and \quad y_{b_i,\min} \le y \le y_{b_i,\max}\} \tag{6.13}$$

where $i = 1, \cdots, N$. Unlike the algorithm of Ha *et al.* using two scalar functions to define the horizontal and vertical projections, we define two variable functions for the horizontal and the vertical projections. The horizontal projection function, $H_{b_i}(x,y)$ is defined by

$$H_{b_i}(x,y) = \begin{cases} x - x_{b_i,\min} + 1 & if \quad y_{b_i,\min} \le y \le y_{b_i,\max} \\ x_{b_i,\max} & if \quad y \ge y_{b_i,\max} \\ 0 & if \quad y \le y_{b_i,\min} \end{cases} \tag{6.14}$$

Likewise, $V_{b_i}(x,y)$, the vertical projection function, is defined by

$$V_{b_i}(x,y) = \begin{cases} y - y_{b_i,\min} + 1 & if \quad x_{b_i,\min} \le x \le x_{b_i,\max} \\ y_{b_i,\max} & if \quad x \ge x_{b_i,\max} \\ 0 & if \quad x \le x_{b_i,\min} \end{cases} \tag{6.15}$$

Two projection tables are created based on the above two projection functions. The horizontal projection table $H_B(x,y)$ is calculated by

$$H_B(x,y) = \sum_{i=1}^{N} H_{b_i}(x,y) \tag{6.16}$$

Likewise, the vertical projection table $V_B(x,y)$ is calculated by

$$V_B(x,y) = \sum_{i=1}^{N} V_{b_i}(x,y) \tag{6.17}$$

For a particular zone *Z*, suppose $\{x_1(Z), y_1(Z)\}$ and $\{x_2(Z), y_2(Z)\}$ are the coordinates of the upper-left and low-right points of the zone. The horizontal and vertical projection files of the zone *Z*, $H_Z(i)$ and $V_Z(j)$, can be computed by the following equations respectively.

$$H_Z(i) = H_B(x_2(Z), i) - H_B(x_1(Z), i), \quad y_1(Z) \le i \le y_2(Z) \tag{6.18}$$

$$V_Z(j) = V_B(j, y_2(Z)) - V_B(j, y_1(Z)), \quad x_1(Z) \le j \le x_2(Z) \tag{6.19}$$

The advantage of our projection functions over Ha *et al.*'s projection functions is that the projection information of an arbitrary zone can be computed using the equation 6.18 and 6.19 deduced from our projection functions. Thus the procedure of projection calculation is separated from the recursive cut procedure. Hence our projection functions can speed up the whole algorithm.

Figure 6.4 outlines the steps of the proposed algorithm of the recursive X-Y cut by the bounding boxes. In the zoned directional X-Y trees generation algorithm for chart images, the starting projection direction for different zones may be different according to the structure relationship of a chart. For instance, the starting directions for the zones of the Y-axis major area and the Y-axis minor area are vertical. While the starting directions

for the zones of the X-axis major area and the X-axis minor area are horizontal. The threshold T_{gap} for areas with vertical direction as the Y-axis area in 2-D charts and the Z-axis area in 3-D charts is equal to half of the median width of the bounding boxes in that area. The threshold T_{gap} for other areas is empirically selected as $T_{gap} = 2$.

1. Generate vertical and horizontal projection tables, $V_B(x, y)$ and $H_B(x, y)$, from a given set of bounding boxes *B* using equations (6.14), (6.15), (6.16), (6.17).
2. Select the sub-root node as the parent node. Select a starting projection direction.
3. Calculate vertical or horizontal projection file $H_Z(i)$ and $V_Z(i)$ using equations (6.18) or (6.19) for the parent node in a specified direction.
4. Split the projection profile into smaller segments at large gaps that exceed a certain threshold T_{gap}. Generate child nodes based on the segments and insert the child nodes under the parent node.
5. Set a child node as a parent node, and alternate the projection direction.
6. Do step 3 to step 5 recursively until there is no splitting needed.

Figure 6.4: Algorithm of Recursive X-Y Cut by the Bounding Boxes

6.2.3 Linking Bounding Boxes with the Zoned Directional X-Y Tree

The leaves of the zoned X-Y tree are usually the bounding boxes or their directional transformed ones of the character-like elements we obtained from connected component classification during pre-processing. But not all the cases satisfy it. For example, there are three bounding boxes for the letter of "%". But these three bounding boxes cannot be found in the zoned X-Y tree due to the intersection between them. Thus insert operations should be applied on the zoned X-Y tree to reflect the correct structural relationship. After recursive X-Y cut procedure, the structural relationship between the bounding boxes and their corresponding location in a document image can be established by linking the bounding boxes with the X-Y tree. Figure 6.5 illustrates the algorithm of linking bounding boxes with the zoned directional X-Y tree.

1. For each set of bounding boxes, find its corresponding sub-root node. Then do step 2 to step 4.
2. For each bounding box, set the sub-root node as the parent node.
3. Search the children nodes of the parent node to find the node that includes or equals to the bounding box. If an equal node is found, link the bounding box with the node, go to step 5. Otherwise, set the found node as the parent node, go to step 3. If no node is found, go to step 4.
4. Generate a new node for the bounding box and insert it under the parent node. Link the bounding box with the new node.
5. Repeat step1 until no unlinked bounding box is left.

Figure 6.5: Algorithm of linking bounding boxes with the zoned directional X-Y tree

6.2.4 Algorithm of Zoned Directional X-Y Tree Generation

In this section, we summarize the algorithm of zoned directional X-Y tree generation.

The whole document image with character-like components is first decomposed into different zones by area separation procedure. In our chart graphical processing, the detected axes can be used as the criterion for area segmentation. Most of our testing charts are 2-D charts. We decompose the 2-D chart image into five zones: the stage area, the X-axis major area, the X-axis minor area, the Y-axis major area and the Y-axis minor area. Since most of our scanned 2-D charts are not skewed seriously, the area segmentation for these charts is simple according to the position of the detected axes. For skewed 2-D charts and 3-D charts, we label the defined areas by hand. A complex area segmentation algorithm with known axes will be one of our future works. The text elements are then categorized into different zones according to their positions.

Figure 6.6 outlines the main processing steps of the whole zoned directional X-Y tree generation algorithm.

1. Decompose the entire image into different zones and insert the zone nodes under the root node.
2. Categorize the text elements into different zones according to their positions.
3. If the orientation of the zone is not horizontal or vertical, apply directional transforms on all the bounding boxes for every zone.
4. Starting from a zone node, if the orientation of the zone is not horizontal or vertical, perform recursive X-Y cut algorithm using the transformed bounding boxes, otherwise use the original bounding boxes.
5. Repeat step 4 until there is no zone node left.
6. Link the projected bounding boxes with the zoned directional X-Y tree.

Figure 6.6: Algorithm of zoned directional X-Y tree generation.

6.3 Text Primitives Labeling

After the zoned directional X-Y tree is generated for the text primitives, a scheme combining X-Y tree searching and traversing with structural analysis is proposed to label the text primitives in a chart. The axes tick labels and titles are the essential elements to change data from a chart representation to a tabular representation.

6.3.1 Extracting Axes Tick Labels

The axes tick labels indicating the measurement for the categories and data values are one of the key elements in interpreting a chart.

The axes tick labels are in the axes major areas. Therefore, starting from the root node of the zoned directional X-Y tree, select the zone node representing a particular axis major area to extract the axes tick labels of a particular axis. Set the zone node as the parent node. The child node whose distance to the axis is the shortest is the area of tick labels for the axis. Set this node as the current processing node. The median height and the median width of the bounding boxes in the area of tick labels are computed. If the median height is greater than the median width, the printing direction for words is in the horizontal direction. Otherwise, the printing direction is in the vertical direction. The axes tick labels are classified into three types: vertical axes tick labels, horizontal axes tick labels and directional axes tick labels. The X-axis tick labels and the Y-axis tick labels in 3-D charts are the directional axes tick labels. Different procedures are applied to extract these three types of tick labels.

Vertical Axes Tick Labels Extraction

The Y-axis tick labels in 2-D charts and the Z-axis tick labels in 3-D charts are the vertical axes tick labels. If the detected axis is in the vertical direction, the X-Y tree of this axis major area is generated by the bounding boxes in the image space. The Y-axis tick labels in skewed 2-D charts are *slant-vertical* axes tick labels. The directional X-Y tree for them is constructed from directional transformed bounding boxes. Yet the extraction procedures for both vertical and slant-vertical axes tick labels are same. In our processing, the printing direction of the vertical axes ticks is in the horizontal direction. Thus if the printing direction of the tick label area is in the horizontal direction, the children of the node corresponding to the area of tick labels are the vertical or slant-vertical axes tick labels.

Horizontal Axes Tick Labels Extraction

The X-axis tick labels in 2-D charts are the horizontal axes tick labels. Likewise, the X-axis tick labels in skewed 2-D charts are *slant-horizontal* axes tick labels. The extraction procedures for both horizontal and slant-horizontal axes tick labels are same. The bounding boxes are represented by the equation (6.13). Sort all the bounding boxes by ordering $x_{b_i,\min}$ in ascending order. The distance between two neighboring bounding boxes b_i and b_{i+1} is calculated using the following equation.

$$D_i = x_{b_{i+1},\min} - x_{b_i,\max} \tag{6.20}$$

There are two spacings, the inter-character spacing and the inter-word spacing for most of the horizontal axes tick labels. But some horizontal axes tick labels have one

spacing, the inter-word spacing. Let $D_{\max}$ and $D_{\min}$ denote the largest distance and the smallest distance respectively. If the following criterion in (6.21) satisfies, then there is one spacing in the horizontal axes tick labels. The children of the node corresponding to the area of tick labels are horizontal or slant-horizontal axes tick labels.

$$\left\| \frac{D_{\max} - D_{\min}}{D_{\min}} \right\| < 1 \tag{6.21}$$

Otherwise, the distance distribution of the bounding boxes is bi-modal. One mode represents the inter-character spacings within words. The other mode represents the inter-word spacings within the line. A threshold can be chosen in the valley between the two peak modes. We use the thresholding technique proposed by Otsu [87] to find an optimum threshold value D_{opt}. The bounding boxes whose neighboring distances are less than the optimum threshold D_{opt} are grouped into the horizontal or slant-horizontal axes tick labels.

Directional Axes Tick Labels Extraction

The extraction of the X-axis tick labels or the Y-axis tick labels in a 3-D chart belongs to this category. Although the directions of the inter-tick-label are along the axes, the direction of the inter-character is in the horizontal direction. The zoned directional X-Y tree for this category is generated based on directional transformed bounding boxes. Starting from the current processing node corresponding to the area of the axes tick labels, extract all the leave nodes by traversing the sub-tree. The original bounding boxes in the image space for the leave nodes are used to extract horizontal lines.

Sort the bounding boxes ordering $y_{b_i,\min}$ in ascending order. Cut the sorted bounding boxes into different groups of axes tick labels at the positions where $y_{b_{i+1},\min}$ is greater than $y_{b_i,\max}$. The output groups are the axes tick labels.

6.3.2 Extracting Titles

Two kinds of titles are extracted in our work: the axes titles and the figure title. Most the axes titles are parallel and close to corresponding axes tick labels. Yet some axes titles for the vertical axes are arranged above the vertical axes.

Starting from the zone node of axis major areas, the child node whose distance to the axis is the second shortest is the title for the axis. If such child node for a vertical axis is not found, search the axes minor area to find the node sharp above the vertical axis as the title of the vertical axis. Most of the figure titles with a large spacing with the axis titles are under the plot area. Starting from the zone node, other non-classified child nodes are the candidates of the figure title. The candidate in the area below the stage area has the highest probability as the title of figure. If the figure title is not found in that area, the candidate in the area above the stage area is considered as the title of the figure.

6.4 Chart Interpretation

The labeled text primitives such as axes tick labels and titles are fed into OCR to obtain textual information. Chart interpretation is to uncover the tabular data from which a chart is generated. In this chapter, we propose a chart interpretation by correlating value points with the axes tick labels.

6.4.1 Chart Interpretation by Correlating Value Points with Tick Labels

Tick labels of the Y-axis normally indicate the value dimension. Tick labels of the X-axis indicate the category dimension. Value points are points that convey the value information of the data marker on a particular position. The horizontal value of the value point indicates the category that the data marker belongs to. The vertical value of the value point indicates the value that the data marker represents.

There may be multiple value points for a data marker at a particular position. For instance, the points on the top line of a bar pattern are the value points of the bar pattern. These value points indicate one unique value of the bar pattern. These value points are named *unique-meaning value points* in our work. Some value points related with a data marker may indicate several values of the data marker. For example, in a high-low-close stock chart, the value points at the high, middle and low position of the line segments represent different value aspects of the high-low-close pattern. These value points are called *multiple-meaning value points.*

In our work, we interpret two kinds of 2-D charts: bar charts and line charts. The value points of these two categories are unique-meaning value points. Since the bar charts represent p comparison and line charts represent trends, the category dimension is discrete for a bar chart and continuous for a line chart.

After chart segmentation, we have segmented bar patterns for bar charts. The transition point farthest to the X-axis of the left side of a bar pattern is the value point of that bar. For a line chart, the transition points for every select feature vector are the value

points. The coordinate positions for both categories of transition points are recorded in the principal variant vectors.

After text primitive analysis, axis tick labels are extracted and bounded by rectangles. The centers of the rectangles in the image represent the coordinate positions of these tick labels. The extracted bounding boxes of tick labels are then passed to Optical Character Recognition (OCR) module to obtain their textual information.

Let $C_1 \ldots C_N$ represent the textual meaning of N category tick labels and $V_1 \ldots V_M$ the textual meaning of M value tick labels. The coordinate positions of the category tick labels and the value tick labels are $C_n(x)$, $C_n(y)$ and $V_m(x)$, $V_m(y)$ respectively, where $1 \le n \le N$ and $1 \le m \le M$. Given a value point P, its coordinate positions in horizontal and vertical direction are $P(x)$ and $P(y)$ respectively. Figure 6.7 illustrates relationship between the value point and tick labels.

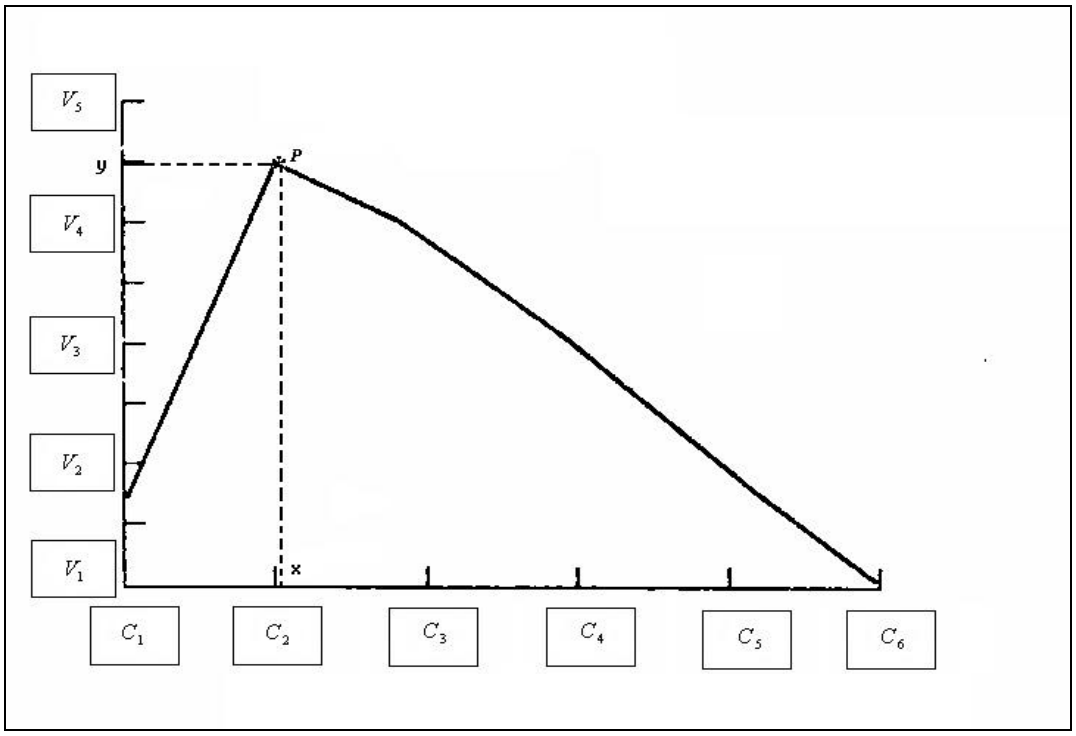


Figure 6.7: Illustration of relationship between a value point and tick labels. *P* is a value point. Tick labels are represented with bounding boxes.

The task of chart interpretation is to calculate the value of the value point, $V(P)$, and the category of the value point, $C(P)$. Suppose the closest range of $C(x)$ for $P(x)$ is between $C_m(x)$ and $C_{m+1}(x)$, and the closest range of $V(y)$ for $P(y)$ is between $V_n(y)$ and $V_{n+1}(y)$. The $V(P)$ is calculated using equation (6.22).

$$V(P) = V_n + \frac{V_{n+1} - V_n}{V_{n+1}(y) - V_n(y)} \times (P(y) - V_n(y)), \quad \textit{and} \quad V_n(y) \le P(y) \le V_{n+1}(y) \quad (6.22)$$

The $C(P)$ in a line chart is calculated using the similar equation (6.23).

$$C(P) = C_m + \frac{C_{m+1} - C_m}{C_{m+1}(x) - C_m(x)} \times (P(x) - C_m(x)), \quad \textit{and} \quad C_m(x) \le P(x) \le C_{m+1}(x) \quad (6.23)$$

The $C(P)$ in a bar chart is calculated using equation (6.24).

$$C(P) = \begin{cases} C_m & |P(x) - C_m| < |C_{m+1} - P(x)| \\ C_{m+1} & Otherwise \end{cases} \quad (6.24)$$

Figure 6.8 outlines the steps of chart interpretation by correlating the value points with the tick labels.

1. Extract the value points from the principal variant feature vectors.
2. For each value point, search its closest ranges in both horizontal and vertical directions.
3. Compute $V(P)$ using equation (6.22).
4. If the chart is a line chart, calculate $C(P)$ using equation (6.23). If it is a bar chart, calculate $C(P)$ using equation (6.24).
5. Repeat step 2 until all the value points are interpreted.
6. Arrange the $V(P)$ and $C(P)$ into two lines and output the tabular data.

Figure 6.8: Chart interpretation by correlating the value points with the tick labels.

The output of chart interpretation is the tabular form of the data. Figure 6.9 shows the interface of the tabular output of chart interpretation. The up corner area of the interface is the display area of the tabular data.

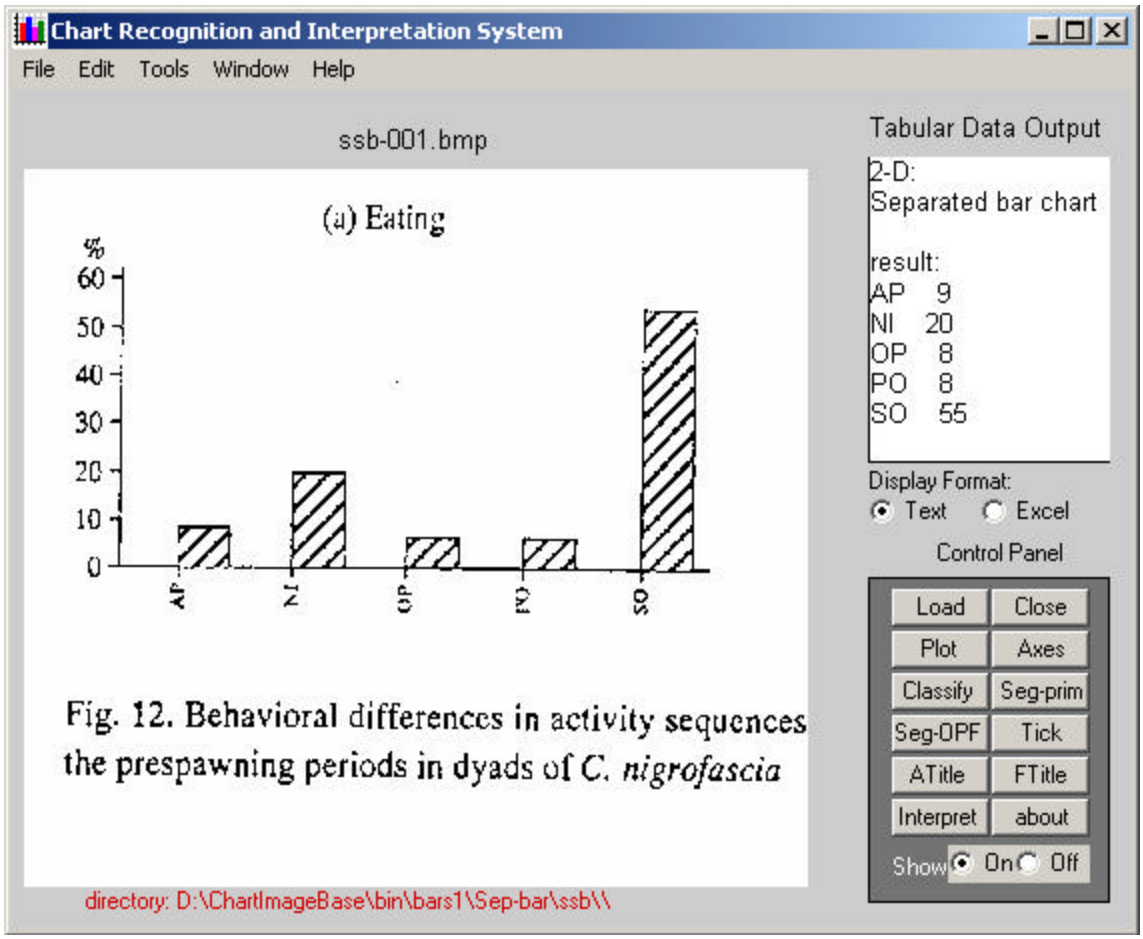


Figure 6.9 Interface of the tabular data output of chart interpretation. Tabular data output is at the top right corner of the interface.

6.5 Experiments and Analysis

The occurrence of the identical layouts of text primitives is frequent in our testing images. We select one text primitive layout among all identical layouts so as to avoid the bias caused by preferring to text primitives layouts with a high performance. The text primitive layouts of 231 2-D images and 16 3-D images are used in our test. The outputs of text primitive labeling are rectangles that represent the smallest area embracing all the text elements in the group.

Figure 6.10 shows the result of extraction and labeling three kinds of text primitives in a 3-D chart including the axes tick labels, the axes titles and the figure title. The figure title is first extracted since it is not in the plot area. After we obtain the axes information from axes detection, the areas along the axes are labeled manually. In each area, apply directional X-Y segmentation and establish a directional X-Y tree for each area. The axes tick labels and axes titles are segmented in subfigures (c) and (d) of figure 6.10 respectively.

Experiments on extraction axes tick labels and titles are reported in the following sections. The precision is defined as the proportion of detected text primitives that is actually correct. The recall measures the probability of correct detected text primitives on existing text primitives.

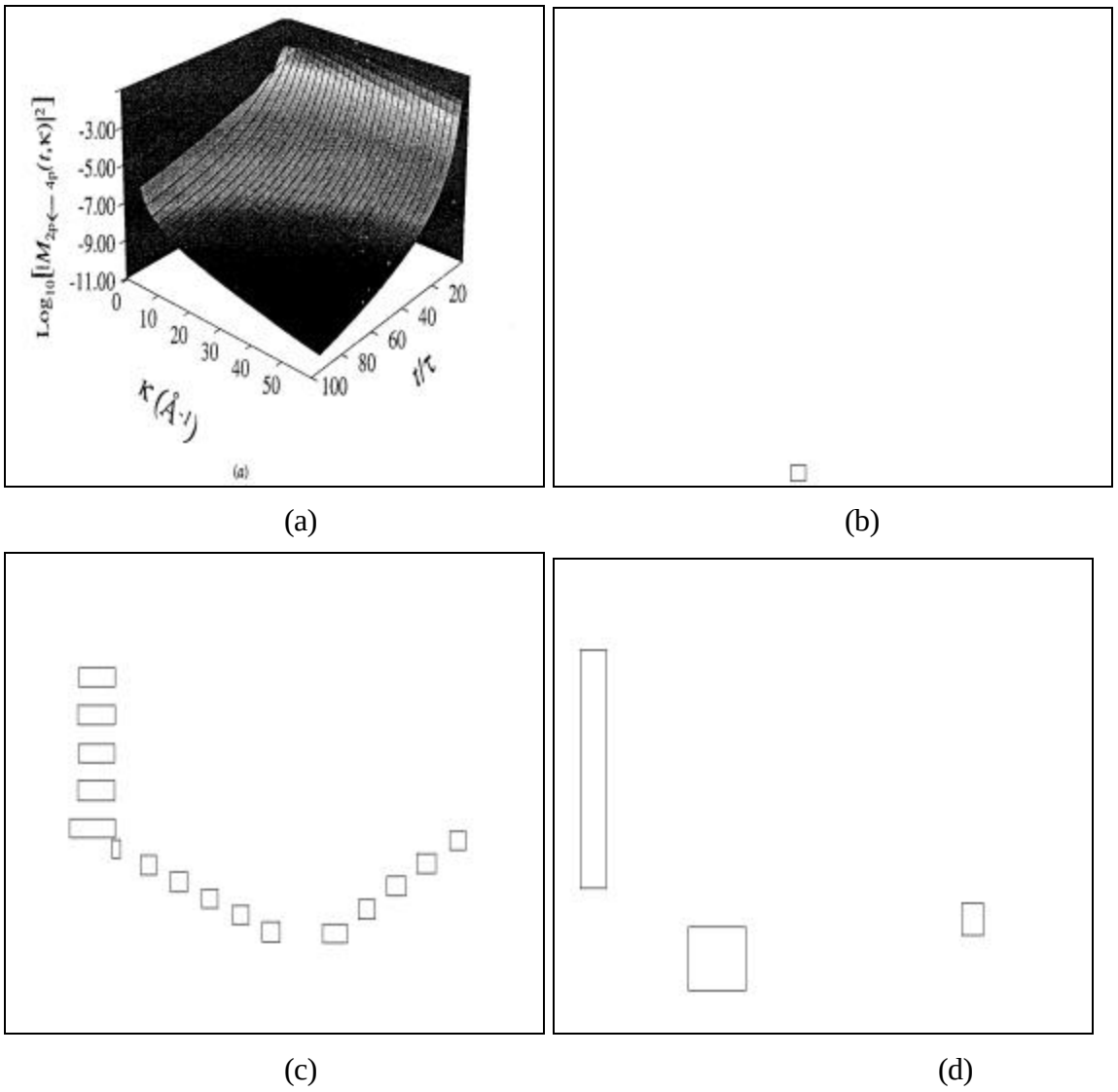


Figure 6.10. The results of text primitive labeling in a 3-D chart. (a). The original 3-D charts. (b) The output of figure title. (c). The output of axes tick labels including vertical and directional axes tick labels. (d). The output of axes title.

6.5.1 Experiments on Axes Tick Labels Extraction

Three kinds of tick labels are extracted including vertical tick label, horizontal tick label and directional axes tick label. X-axis and Y-axis tick labels with slant orientation in 2-D charts are categorized as directional axes tick labels. The extraction vertical and horizontal axes tick labels uses the original bounding boxes in the recursive X-Y cut algorithm. Table 6.1 shows the result of these two classes of axes tick label extraction.

Axes tick labels	Total	Detected	Correct	Precision	Recall
Vertical	859	843	772	91.58%	89.87%
Horizontal	962	918	884	96.3%	91.9%

Table 6.1: Results of vertical and horizontal axes tick labels extraction

In most of our testing 2-D charts, the distribution of the axes labels in the axes is sparse and the axes labels can be easily extracted. But in some charts, the axes labels are very close to the axes titles or even overlap with the axes title by some characters such as "y", "g", etc or by the subscripts and superscripts in the axes title such as "$x^2 = d$ ", "c_y ", etc. Most of such situations can be found in the area around the vertical axes where the characters in the tick labels are arranged in a horizontal direction while the characters in the axes title are arranged in a vertical direction. Thus the overall performance of the vertical axes tick label extraction is worse than that of the horizontal axes tick label extraction.

The directional axes labels that include X-axis and Y-axis tick labels with slant orientation in some 2-D charts and directional axes labels in 3-D charts are extracted

using directional transformed bounding boxes in our work. The extraction using the original bounding boxes without directional transform cannot be applied to detect those directional axes labels because it cannot correctly group the area of the directional axes labels. Table 6.2 reports the performance of experiments on directional axes tick labels extraction.

Axes tick labels	Total	Detected	Correct	Precision	Recall
Directional axes tick labels	126	113	93	82.30%	73.81%

Table 6.2: Results of directional axes tick labels extraction.

From table 6.2, we can see that the directional axes tick labels can be extracted by using directional X-Y tree segmentation with transformed bounding boxes. Some failed extractions are due to the overlapping structure in the alternating directions. Although the directional X-Y tree is a generalized version of the original X-Y tree structure, like the original X-Y tree structure, the directional X-Y tree also requires strict tiling of text blocks in different orientations. In the future work, the domain knowledge such as the structural relationship for specific type of charts or the language knowledge such as the result of OCR can be use to post-process the results from the directional X-Y segmentation in order to achieve a higher accuracy.

6.5.2 Experiments on Titles Extraction

The axes titles including the X-axis and the Y-axis titles in 2-D charts and the X-axis, the Y-axis and the Z-axis titles in 3-D charts are extracted. Not all the testing charts have

figure title or axes titles. The results are reported in table 6.3. The extraction of the figure titles is also reported in table 6.3.

Titles	Total	Detected	Correct	Precision	Recall
Axes titles	348	316	291	92.09%	83.62%
Figure titles	186	176	171	97.16%	91.93%

Table 6.3: Results of axes titles and figure titles extraction

Most of the figure title can be extracted correctly. Some figure titles are positioned in the area other than axis-major areas such as the stage areas, etc. These figure titles are misclassified as other labels such as legends, data values, etc. The positional variation for the axes titles is usually around the axis. But axes title extraction is normally influenced by the results of the axes tick label extraction that are closer to the axes titles comparing with the figure title. Yet in most charts, these text primitives are exclusive, i.e., if one primitive is labeled as the figure title then the other primitive cannot be labeled as the figure title either. The language knowledge such as a chart dictionary with common chart words such as “%”, “Fig.”, etc, can be a good help to improve the overall performance of the text primitive extraction.

6.6 Summary

A zoned directional X-Y tree structure is proposed to hierarchically represent the text primitives in charts. An algorithm of generating the zoned directional X-Y tree is presented. The algorithm includes three procedures: directional transformation of the bounding boxes, recursive X-Y cut by the bounding boxes and linking the bounding boxes with the X-Y tree. A scheme combining X-Y tree searching and traversing with structural analysis is proposed to label the text primitives in a chart. Three kinds of axes tick labels are extracted: vertical axes tick labels, horizontal axes tick labels and directional axes tick labels. The extraction of the axes titles and the figure titles is also presented. Experiments and analysis for extraction axes tick labels and titles are presented. A simple method by correlating the value points with the tick labels in interpreting bar charts and line charts is proposed. The final output for chart interpretation is a text file or an Excel file that illustrates the value and category of data markers and other chart information.

The contributions for this chapter are as follows:

1. We propose a zoned directional X-Y tree structure to hierarchically represent the text in graphical documents. The proposed zoned directional X-Y tree is a generalized version of the classical X-Y tree which considers only orientations in the vertical and the horizontal directions.
2. A method of directional transforming the bounding boxes in the image space to the r-space is proposed.

3. A recursive X-Y cut segmentation algorithm using original and transformed bounding boxes is proposed to generate the zoned directional X-Y tree for text primitives.
4. We present an approach of combining X-Y tree searching and traversing with structural analysis to label the text primitives in a chart. Detailed procedures to extract axes tick labels and titles are illustrated.
5. We propose an approach to interpret chart by correlating the value points with the tick labels.

.

Chapter 7

Future Directions and Conclusion

In this final chapter, we conclude the dissertation with a discussion of the future work relating with chart recognition.

7.1 Future Directions

7.1.1 Broadening Chart Types for Model-based Chart Classification

In this dissertation, we have constructed four general chart models for bar charts and line charts in the proposed framework of model-based chart classification. Although bar and line charts are the most commonly used chart representations to present discrete comparison and continuous trend, other types of charts such pie charts, data charts and stock charts are also very commonly used. Future efforts should focus especially on the stock charts, the most frequently used charts in the stock markets. Combination charts where two or more chart types are mixed together or color charts where color conveys important discrimination information are also the future focus in model-based chart classification. How to construct a model-based chart classification that can discriminate a wide range of different chart types?

One of the solutions for it is to develop a more sophisticated feature extraction scheme. We may expand the feature points by investigating the domain knowledge of different charts. More features capturing essential discriminate information may be added in the feature vector, such as color feature. More coarse-to-fine procedures may be needed to segment appropriate feature points and features.

Another solution for it lies in the chart model construction. How to select an appropriate chart model and topology to represent a chart type is important in chart model construction.

7.1.2 More Label Types in Text Primitive Labeling

In this dissertation, we have constructed a zoned directional X-Y tree to organize the text primitives in a chart. Three types of labels, axis tick mark labels, axis titles and figure titles, are recognized by traversing the established X-Y tree in text primitive labeling. While these three types are essential elements in chart interpretation, other labels such as legends, also convey important information for chart understanding and interpretation.

7.1.3 Integrating Low-Level Heuristic Search with Optimal Path Finding for Chart Segmentation

Both low-level heuristic search approach and optimal path finding approach are proposed for chart segmentation in our dissertation. Each has its own advantages and limitations. While low-level heuristic search can provide precise local information from uncomplicated chart images, the performance of it may deteriorate a lot in complicated chart images such as charts with textured patterns or patterns with 3-D effect. Optimal

path finding approach is a statistical method trained on a large amount of data. The underlying optimal path may not represent the precise local information in chart segmentation. Thus it can be promising if we can combine the low-level heuristic features into optimal path finding appropriately.

7.1.4 Exploring Complex Feedback Mechanism

In our work, the text processing and graphics processing are comparatively independent of each other although the text processing does utilize the orientation information provided by the graphics processing. Exploring complex feedback mechanism to refine the result of both processing procedures without contradiction with the argument of comparative independence will be instrumental in improving the robustness of the chart recognition system.

7.1.5 Integrating More Knowledge Sources for Chart Recognition and Interpretation

In the dissertation, we have integrated some geometric knowledge and domain knowledge when we detect the axes in a chart, extract features for model-based classification, segment the chart by low-level heuristic search and interpret a bar chart or a line chart.

If more types of chart are to be recognized, the domain knowledge about those charts can be helpful in chart recognition and interpretation. For instance, the high point, the low point and the middle point of a high-low-close bar represent three different levels of value for high-low-close chart recognition.

Other examples of knowledge sources include:

- Syntactical constraints.

  The structural organization of chart labels follows some syntactical constraints. For example, if both axis title and figure title exist in a chart, the axis title is closer to its corresponding axis.

- Perceptual knowledge. For example, the continuity of a line may be helpful in separating multiple lines at the crossing point.

- Language knowledge. For example, chart dictionary including words such as "%", "Fig.", can be helpful in discriminating different text labels.

7.2 Conclusion

In this dissertation, we have investigated four problem domains in chart recognition: chart recognition system, chart graphic symbol extraction, chart classification and segmentation, text primitive analysis and chart interpretation. The following are our contributions in these four problem domains, in answer to the targets of contribution set out in chapter 1.

Chart recognition system: We proposed a hierarchical statistical-model-based framework for scientific chart recognition system. In order to support the rationality of the proposed framework, first, the knowledge of chart generation software has been explored and notation conventions of a scientific chart from both a generation point of view and recognition point of view have been defined. Second, investigation in psychological aspect and human visual perception on charts deduces three arguments that are the principles and backbone of proposed framework. The main parts of our chart

recognition system include preprocessing, chart graphics symbol recognition, chart classification and segmentation, text primitive analysis and chart interpretation. Our testing data is constructed with more than 500 chart images from technical journals that are scanned at 300 dpi.

Chart graphic symbol extraction: Preprocessing includes common procedures such as noise removal, connected component analysis and component classification. Chart graphics symbol recognition of current work includes plot area detection and axis detection. We proposed an improved projection-based plot area detection method which shows the highest recall rate by comparing with our previous method and others' method. For axis detection, we present a Hough-based axis detection algorithm to discriminate X-axis, Y-axis and Z-axis. Experiments show our method is more accurate in extracting axes and their position information comparing with other projection-based axes detection method.

Chart classification and segmentation: We proposed a new approach for chart classification and segmentation based on statistical modeling. Four chart models including separated bar model, contiguous bar model, single-line-series line model and multiple-line-series line model are constructed and trained using a segmental *K*-means algorithm to model the semantics of chart stage area. Charts are classified by choosing the chart model with the largest posteriori probability. The best state path for that model is also obtained by applying Viterbi algorithm. Two kinds of classifications, dimension classification and type classification, are addressed. We also proposed a new approach for chart segmentation using optimal path finding. Two chart segmentation problems are addressed, including detecting the number of data series and bar pattern segmentation.

Text primitive analysis and chart interpretation: We proposed a zoned directional X-Y tree structure to hierarchically represent the text primitives in charts. An algorithm of generating the zoned directional X-Y tree is presented. The algorithm includes three procedures: directional transformation of the bounding boxes, recursive X-Y cut by the bounding boxes and linking the bounding boxes with the X-Y tree. A scheme combining X-Y tree searching and traversing with structural analysis is proposed to label the text primitives in a chart. Three kinds of axes tick labels are extracted: vertical axes tick labels, horizontal axes tick labels and directional axes tick labels. The extraction of the axes titles and the figure titles is also presented. Experiments and analysis for extraction axes tick labels and titles are presented.

Finally, both results from graphics processing and text primitive analysis are correlated for chart interpretation. The final output for chart interpretation is a text file or an Excel file that illustrate the value and category of data markers and other chart information after the text primitives are recognized by OCR.

Appendices

A Hough Transform

The principle of the Hough transform is to detect geometric shapes by utilizing a voting mechanism to estimate parameters which represent different shape such as line, circle, etc. Let us consider this approach for finding a straight line. The following polar form is normally adopted to represent a straight line.

$$r = x\cos q + y\sin q \tag{A.1}$$

Unlike the slope-intercept parameterization, this parameterization gives a bounded parameter space. Figure A-1 illustrates the mechanism of Hough transform. Each point in the parameter space is a line in the feature space. Each point in the feature space is a curve in the parameter space. The value of each point in the parameter space indicates the number of points lying on the line. Hough transform is a mature technique for shape detection. The algorithm of standard Hough transform can be found in many image processing or computer vision books such as [52]. Unlike the explanation of the algorithm mentioned in [52] which focuses on the implementation of the algorithm, we propose our explanation for this algorithm by using algebra as follows.

Both r and q need to be quantized in order to construct a two-dimensional accumulator array in the r-q plane. Suppose the r-q space for $-R \le r \le R, 0 \le q \le p$ is quantized into an $r \times c$ accumulator array. The increment (Δq) in the angle q is $\frac{p}{c}$. The increment (Δr) in r is $\frac{2R}{r}$. The accumulator image $H(i, j)$ is defined over Y, where

$$Y = \{(r_i, q_j) : r_i = (i - \frac{r}{2})\frac{2R}{r}, \quad q_j = \frac{jp}{c}, 0 \le i \le r, \quad 0 \le j \le c \ \} \tag{A.2}$$

Suppose there are N feature pixels in a source image. For a random feature pixel(x_k, y_k), its accumulator image function $h(i, j)_k$ is computed as

$$h(i, j)_k = \begin{cases} 1 & if \ | \ r_i - (x_k \cos q_j - y_k \sin q_j) | < \frac{p}{2c} \\ 0 & otherwise \end{cases} \tag{A.3}$$

Thus the accumulator image is computed by

$$H(i, j) = \sum_{k=1}^{N} h(i, j)_k \tag{A.4}$$

The Hough transform can be generalized to detect arbitrary shapes which can be described by multiple parameters. More considerations on the topic of generalized Hough transform can be found in [52, 66].

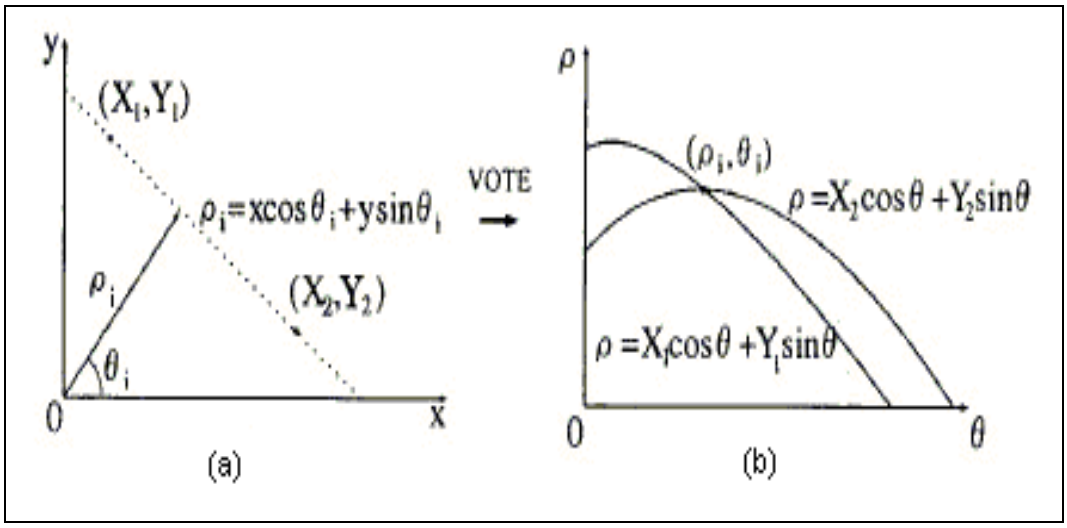


Figure A.1. The mechanism of Hough transform: (a): two points (x_1, y_1) and (x_2, y_2) are on the same line in the image space (X-Y). (b): the points (x_1, y_1) and (x_2, y_2) are transformed into two curves in the parameter space (r -q). The coordinates of the crossing point of the curves are the parameters of the line.

B Hidden Markov Models

A hidden Markov model is a two-stochastic-variable process, in which one of the variables is hidden, and the other random variable is observable. The hidden random variable X_N represents the states, in a model that is a collection of states connected by transitions. O_T denotes the observation variable. Most of the notation and material presented in this section are adapted from Rabiner [91, 92]. We denote the following model notation for a discrete first order hidden Markov Model.

T	Length of the observation sequence.
N	Number of states in the model.
M	Number of observation symbols.
S	$\{ S_1, S_2, \ldots, S_N \}$, states.
Q	$\{ x_1 x_2 \ldots x_T \}$, state sequence.
V	$\{ v_1, v_2, \ldots, v_M \}$, discrete emitting symbol set of possible observation.
x_t	State visited at time t.
A	$\{ a_{ij} \}$, state transition probability distribution.
B	$\{ b_j(k) \}$, output observation probability distribution.
p	$\{ p_i \}$, initial state distribution.

The state sequence includes an initial state x_1 and a final state x_T. Each state has two sets of probabilities:

The Output Observation Probability

It provides the conditional probability that being in a particular state, a symbol is generated from a finite set of symbols. The probability of emitting symbol v_k being in state S_i at time t is:

$$b_i(k) = P(O_t = v_k \mid x_t = S_i) \tag{B.1}$$

where $x_t = S_i$ means the Markov chain was in state S_i at time t, and $O_t = v_k$ means the output symbol at time t was v_k.

The Transition Probability

It defines the probability of taking the transition from a particular state to another state or itself. It is calculated using the following equation:

$$a_{ij} = P(x_{t+1} = S_j \mid x_t = S_i) \tag{B.2}$$

where $x_t = S_i$ means the Markov chain was in state S_i at time t, and $x_{t+1} = S_j$ means the Markov chain was in state S_j at time $t+1$.

The initial state distribution probability is defined by

$$p_i = P(x_1 = S_i) \tag{B.3}$$

Two assumptions are used in the first order hidden Markov Model. The first assumption is that if the Markov model will be in a particular state S_j at time $t+1$, it only depends on the present state S_i at time t, and not in the past states. The following equation is satisfied.

$$P(x_{t+1} = S_j \mid x_t = S_i, ..., x_1 = S_1) = P(x_{t+1} = S_j \mid x_t = S_i) \quad \text{(B.4)}$$

Another assumption is that an observed symbol v_k only depends on the present state at time t, and not in the past states or emitted observations. For a give sequence $O = O_1, O_2, ..., O_T$, the assumption can be expressed as the following equation.

$$P(O_t = v_k \mid O_{t-1} = v_{k-1}, ..., O_1 = v_1, x_t = S_i, ..., x_1 = S_1) = P(O_t = v_k \mid x_t = S_i) \quad \text{(B.5)}$$

The compact notation $l = (A, B, p)$ is normally used to represent a Hidden Markov Model.

There are three main problems to solve using Hidden Markov Models [91].

1. The Evaluation Problem

Given a model $l = (A, B, p)$, compute the probability that it will generate a sequence $O = O_1, O_2, ..., O_T$. A forward-backward procedure is usually applied on solving problems in this category [91].

2. The Optimal-State Sequence Problem

Given a model $l = (A, B, p)$ and a sequence of observations $O = O_1, O_2, ..., O_T$, infer the most likely state sequence in the model that produced the observations. The Viterbi algorithm [117] which is based on dynamic programming solves this category of problems.

3. The Learning Problem

Given a model l and a set of observations, how to train parameters of the model so that the model has a high probability of generating the observations? The Baum-Welch algorithm solves this category of problems [91].

More details on the solutions for the three problems can be found in [49, 91, 92, 117].

Bibliography

[1]. C. Ah-Soon. A constraint network for symbol detection in architectural drawings. In K. Tombre and A. Chhabra, editors, *Graphics Recognition: Algorithms and Systems*, LNCS 1389: 80-90, Springer, 1998.

[2]. A. Amano and N. Asada. Graph grammar based analysis system of complex table form document. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition*, pages: 916-920, 2003.

[3]. H. Baird, H. Bunke and K. Yamamoto, editors. *Structured Document Image Analysis*. Springer Verlag, 1992.

[4]. W. Barrett and H. Nielson. Consensus-based table form recognition. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition*, pages: 906-910, 2003.

[5]. N. Bennett, R. Burridge and N. Saito. A method to detect and characterized ellipses using the Hough transform. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(7): 652 -657, 1999.

[6]. J.R. Bergen and H. Shvaytser. A probabilistic algorithm for computing Hough transforms. In *Journal of Algorithms*, 12(4): 639-656, 1991.

[7]. D. Blostein. General diagram recognition methodologies. In R. Kasturi and K. Tombre, Editors, *Graphics Recognition: Methods and Application*, LNCS 1072:106-122, Springer, 1996.

[8]. D. Blostein and H. Baird. A critical survey of music image analysis. In H. Baird, H. Bunke, and K. Yamamoto, editors, *Structured Document Image Analysis*, pages: 405-434, Springer Verlag, 1992.

[9]. D. Blostein and L. Haken. Using diagram generation software to improve diagram recognition: a case study of music notation. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(11): 1121 –1136, 1999.

[10]. D. Blostein, E. Lank and R. Zanibbi. Treatment of diagrams in document image analysis. In M. Anderson, P. Cheng and V. Haarslev, editors, *Theory and Application of Diagrams*, LNCS 1889, pages: 330-344, Springer-Verlag, 2000.

[11]. H. Bunke, P.S.P. Wang and H. Baird, editors. *Document Image Analysis*, World Scientific, 1994.

[12]. H. Bunke and P. Wang, editors. *The Handbook of Character Recognition and Document Image Analysis*, World Scientific, 1997.

[13]. D. Burr. Elastic matching of line drawings. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 3:708-713, 1981.

[14]. R. Casey, S. Boyer, P. Healey, A. Miller, B. Oudot and K. Zilles. Optical recognition of chemical graphics. In *Proc. of the Second Int. Conf. on Document Analysis and Recognition*, pages: 627-631, 1993.

[15]. F. Cesarini, M. Gori, S. Marinai and G. Soda. A hybrid system for locating and recognizing low level graphic items. In R. Kasturi and K. Tombre, Editors, *Graphics Recognition: Methods and Application*, LNCS 1072:134-147, Springer, 1996.

[16]. F. Cesarini, M. Gori, S. Marinai and G. Soda. Structured document segmentation and representation by the modified X-Y tree. In *Proc. of Fifth Int'l Conf. on Document Analysis and Recognition*, pages: 563–566, 1999.

[17]. K-F. Chan and D-Y. Yeung. Mathematical expression recognition: a survey. In *Int. Journal of Document Analysis and Recognition*, 3(1):3–15, 2000.

[18]. K-F. Chan and D-Y. Yeung. Error detection, error correction and performance evaluation in on-line mathematical expression recognition. In *Pattern Recognition*, Vol. 34: 1671-1684, 2001.

[19]. M.-T. Chang and S.-Y. Chen. Deformed trademark retrieval based on 2D pseudo-hidden Markov model, In *Pattern Recognition*, 34(5): 953-967, 2001.

[20]. M. Chen. Off-line handwritten word recognition using hidden Markov models. In *Proc. of United States Postal Service Advanced Technology Conference*, pages: 563-579, 1992.

[21]. M.-Y. Chen and A. Kundu. An alternative to variable duration HMM in handwritten word recognition. In *Proc. of Int. Workshop on Frontiers Handwriting Recognition*, pages: 48-54, 1993.

[22]. F.-H. Cheng, W.-H. Hsu and M.-Y. Chen. Recognition of handwritten Chinese characters by modified Hough transform techniques. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(4): 429 -439, 1989.

[23]. T. Cheng, J. Khan, H. Liu and Y. Yun. A symbol recognition system. In *Proc. of Second IAPR Int. Conf. on Document Analysis and Recognition*, pages: 918-921, 1993.

[24]. A.K. Chhabra. Graphic symbol recognition: an overview. In K. Tombre and A. Chhabra, editors, *Graphics Recognition: Algorithms and Systems*, LNCS 1389: 68-79, Springer-Verlag, 1998.

[25]. P. Chou. Recognition of equations using a two-dimensional stochastic context-free grammar. In *Proc. SPIE Conf. on Visual Communications and Image Processing IV*, Philadelphia PA, pages: 852-863, Nov. 1989.

[26]. S. Chowdhury, S. Mandal, A. Das and B. Chanda. Automated segmentation of math-zones from document images. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition*, pages 755-759, 2003.

[27]. S.J. Cox. Hidden Markov Models for automatic speech recognition: theory and application. In C. Wheddon and R. Linggard, editors, *Speech and Language Processing*, pages: 209-230, 1990.

[28]. D. Doermann, E. Rivlin, and I. Weiss. Applying algebraic and differential invariants for logo recognition. In *Machine Vision and Applications*, 9(2):73–86, 1996.

[29]. D. Dori. Syntactic and semantic graphics recognition: the role of the object-process methodology. In A.K. Chhabra and D. Dori, editors, *Graphics Recognition: Recent Advances*, LNCS 1941:277-287, Springer, 2000.

[30]. R.O. Duda and P.E. Hart. Use of the Hough transform to detect lines and curves in pictures. In *Communications of the Association for Computing Machinery*, 15:11–15, 1972.

[31]. A. Etemadi, J.-P. Schmidt, G. Matas, J. Illingworth, J. Lladós, J. López-Krahe and E. Martí. A system to understand hand-drawn floor plans using subgraph

isomorphism and Hough transform. In *Machine Vision and Applications*, 10(3):150–158, 1997.

[32]. H. Fahmy and D. Blostein. A graph-rewriting paradigm for discrete relaxation application to sheet-music recognition. In *Int. Journal of Pattern Recognition and Artificial Intelligence*, 12(6): 763-799, 1998.

[33]. C. Fahn, J. Wang and J. Lee. A topology-based component extractor for understanding electronic circuit diagrams. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(2): 1140 –1157, 1989.

[34]. K.-C. Fan, J.-M. Lu and G.-D. Chen. A feature point clustering approach to the recognition of form documents. In *Pattern Recognition*, 31(9):1205-1220, 1998.

[35]. L.A. Fletcher and R. Kasturi. A robust algorithm for text string separation from mixed text/graphics images. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 10(6): 910 –918, 1988.

[36]. E. Francesconi, P. Frasconi, M. Gori, S. Marinai, Q. Sheng, G. Soda and A. Sperduti. Logo recognition by recursive neural networks. In K. Tombre and A. Chhabra, editors, *Graphics Recognition: Algorithms and Systems*, LNCS 1389: 104-117, Springer, 1998.

[37]. R.P. Futrelle and N. Nikolakis. Diagram analysis using context-based constraint grammars. In *Technical Report NU-CCS-96-01*, Northeastern University, pages: 1-19, 1996.

[38]. R. Futrelle, M. Shao, C. Cieslik and A. Grimes. Extraction, layout analysis and classification of diagrams in PDF documents. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition*, pages: 1007-1014, 2003.

[39]. A. Gillies. Cursive word recognition using hidden markov models. In *Proc. of United States Postal Service Advanced Technology Conference*, pages: 557-563, 1992.

[40]. L. O'Gorman, Basic techniques and symbol-level recognition--an overview. In R. Kasturi and K. Tombre, Editors, *Graphics Recognition: Methods and Application*, LNCS 1072: 1-12, Springer, 1996.

[41]. L. O'Gorman and R. Kasturi. *Document Image Analysis*, IEEE Computer Society Press, 1995.

[42]. J. Ha, R. M. Haralick and I. T. Phillips. Recursive X-Y cut using bounding boxes of connected components. In *Int. Conf. on Document Analysis and Recognition*, vol. 2, pages: 952-955, 1995.

[43]. R.M. Haralick. Performance evaluation of document image algorithms. In A.K. Chhabra and D. Dori, editors, *Graphics Recognition: Recent Advances*, LNCS 1941:315-323, Springer, 2000.

[44]. J. D. Hartog, T. T. Kate and J. Gerbrands. Knowledge-based segmentation for automatic map interpretation. In R. Kasturi and K. Tombre, Editors, *Graphics Recognition: Methods and Application*, LNCS 1072: 159-178, Springer, 1996.

[45]. Y. He and A. Kundu. 2-D shape classification using hidden Markov model. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(11): 1172 –1184, 1991.

[46]. T. Henderson and L. Swaminathan. Symbolic pruning in a structural approach to engineering drawing analysis. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition*, pages 180-184, 2003.

[47]. P.V.C. Hough. Method and means for recognizing complex patterns. Technical report, 1962.U.S. Patent No. 3069654.

[48]. J. Hu, M. Brown and W. Turin. HMM based on-line handwriting recognition. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(10): 1039-1045, 1996.

[49]. X.D. Huang, Y. Ariki and M. Jack. *Hidden Markov Models for Speech Recognition*. Edinburgh University Press, 1990.

[50]. M. Hurst. A constraint-based approach to table structure derivation. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition*, pages: 911-915, 2003.

[51]. J. Illingworth and J. Kittler. A survey of the Hough Transform. In *Computer vision, graphics and image processing*, 44: 87-116, 1988.

[52]. R. Jain, R. Kasturi and B. G. Shunck. *Machine Vision*, MIT Press and McGraw-Hill, 1995.

[53]. S. Joseph and T. Pridmore. Knowledge-directed interpretation of mechanical engineering drawings. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(9): 928-940, 1992.

[54]. B.H. Juang and L.R. Rabiner. Mixture autoregressive hidden Markov models for speech signals. In *IEEE Trans. on Acoustic, Speech, Signal Processing*, 33: 1404-1413, 1985.

[55]. D.-M. Jung, G. Nagy and A. Shapira. N-tuple features for OCR revisited. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(7): 734-745, 1996.

[56]. A. Kacem, A. Belaid, and M. B. Ahmed. Automatic extraction of printed mathematical formulas using fuzzy logic and propagation of context. In *Int. Journal of Document Analysis and Recognition*, 4(2):97–108, 2001.

[57]. R. Kasturi, W. El-masri, J. Shah, J.R. Gattiker and U.B. Mokate. A system for interpretation of line drawing. In *IEEE Trans. on Pattern Analysis and Machine Intelligence,* 12(10): 978 –992, 1990.

[58]. R. Kasturi and H. Luo. Research advances in graphics recognition: an update. In *Advances in Document Image Analysis, Proceedings of the First Brazilian Symposium (BSDIA '97),* LNCS 1339:99-110, 1997.

[59]. H. Kato, S. Inokuchi. The recognition method for roughly hand-drawn logical diagrams based on utilization of multi-layered knowledge. In *Proc. 10th Int. Conf. on Pattern Recognition*, pages: 443-473, 1990.

[60]. J. Kittler and J. Illingworth. Minimum error thresholding. In *Pattern Recognition,* 19(1): 41-47, 1986.

[61]. G.E. Kopec and P.A. Chou. Document image decoding using Markov source models**.** In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(6): 602 – 617, 1994.

[62]. G.E. Kopec, P.A. Chou and D. Maltz. Markov source models for printed music decoding**.** In *Journal of Electronic Imaging*, 5(1): 7-14, 1996.

[63]. M. Krishnamoorthy, G. Nagy, Seth, Sharad and M. Viswanathan. Syntactic segmentation and labeling of digitized pages from technical journals. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(7): 737 –747, 1993.

[64]. J. Kupiec, Robust part-of-speech tagging using a hidden Markov model. In *Computer Speech and Language,* 6:225-242, 1992.

[65]. E. Lank. A retargetable framework for interactive diagram recognition. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition,* pages: 185-189, 2003.

[66]. V.F. Leavers. *Shape Detection in Computer Vision Using the Hough Transform*. Springer-Verlag, 1992.

[67]. S. Lee. Recognizing hand-written electrical circuit symbols with attributed graph matching, *Structured document analysis*, pages: 340-358, 1992.

[68]. S.E. Levinson, L.R. Rabiner, and M.M. Sondhi. An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition. In *The Bell System Technical Journal*, pages: 1035-1073, 1983.

[69]. W. –Y. Liu and D. Dori. Genericity in graphics recognition algorithms. In K. Tombre and A. Chhabra, editors, *Graphics Recognition: Algorithms and Systems*, LNCS 1389:9-21, Springer, 1998.

[70]. C.-L. Liu, H. Sako and H. Fujisawa. Performance evaluation of pattern classifiers for handwritten character recognition. In *International Journal on Document Analysis and Recognition*, 4(3): 191-204, 2002.

[71]. J. Llados, J. Lopez-Krahe, and E. Marty. A system to understand hand-drawn floor plans using subgraph isomorphism and Hough transform. In *Machine Vision and Applications*, 10(3):150–158, 1997.

[72]. J. Llados, E. Valveny, G. Sanchez and E. Marti. Symbol recognition: current advances and perspectives. In D. Blostein and Y.-B. Kwon, editors, *Graphics Recognition: Algorithms and Applications*, LNCS 2390: 109-128, Springer, 2002.

[73]. D. Lopresti and G. Nagy. A tabular survey of automated table processing. In A.K. Chhabra and D. Dori, editors, *Graphics Recognition: Recent Advances*, LNCS 1941:93-120, Springer, 2000.

[74]. Y. Luo and W. Liu. Engineering drawings recognition using a case-based approach. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition*, pages: 190-194, 2003.

[75]. Z.-Y. Lu. Detection of text regions from digital engineering drawings. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(4): 431-439, 1998.

[76]. S. Mandal, S. Chowdhury, A. Das, and B. Chanda. Automated detection and segmentation of table of contents page from document images. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition*, pages: 398-402, 2003.

[77]. B. T. Messmer and H. Bunke. Automatic learning and recognition of graphical symbols in engineering drawings. In R. Kasturi and K. Tombre, Editors, *Graphics Recognition: Methods and Application*, LNCS 1072: 123-134, Springer, 1996.

[78]. *Microsoft Excel Help: Charts.* Copyright by Microsoft Corporation, 1985-2002.

[79]. J. Mitra, U. Garain, B. Chaudhuri, T. Pal, and H. Swamy. Automatic understanding of structures in printed mathematical expressions. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition*, pages: 540-544, 2003.

[80]. M. A. Mohamed and P. Gader. Generalized Hidden Markov Models-part II: application to handwritten word recognition. In *IEEE Trans. on Fuzzy Systems*, 8(1): 82-94, 2000.

[81]. I. Mulder, A. Mackworth and W. Havens. Knowledge structuring and constraint satisfaction: the Mapsee approach. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 10(6): 866 –879, 1988.

[82]. G. Nagy. Twenty years of document image analysis in PAMI. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(1): 38–62, 2000.

[83]. G. Nagy and S. Seth. Hierarchical representation of optically scanned documents. In *Proc. of Seventh Int. Conf. on Pattern Recognition*, pages: 347–349, 1984.

[84]. Y. Nakamura, M. Takahashi, M. Onda and Y. Ohta. Knowledge extraction from diagram and text for media integration. In *Proc. of Int. Conf. on Multimedia*, pages: 488 –492, 1996.

[85]. A. Okazaki, T. Kondo, K. Mori, S. Tsunekawa and E. Kawamoto. An automatic circuit diagram reader with loop-structure-based symbol recognition. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 10(3): 331 –341, 1988.

[86]. T. Oshitani and T. Watanabe. Parallel map recognition with information propagation mechanism. In *Proc. of the Fifth Int. Conf. on Document Analysis and Recognition*, pages: 717-720, 1999.

[87]. N. Otsu. A threshold selection method form gray-level histograms. In *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. SMC-9, No. 1, pages: 62-66, 1979.

[88]. D. Pao, H.F. Li, and R. Jayakumar. Shapes recognition using the straight line Hough transform: Theory and generalization. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(11):1076–1089, 1992.

[89]. I.T. Phillips, S. Chen and R.M. Haralick. CD-ROM English document database standard. In *Proc. 2nd Int. Conf. on Document Analysis and Recognition*, pages: 478-483, 1993.

[90]. I.T. Phillips and A.K. Chhabra. Empirical performance evaluation of graphics recognition systems. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(9): 849 –870, 1999.

[91]. L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Proc. of the IEEE*, 77:257-286, 1989.

[92]. L. R. Rabiner and B. H. Juang. An introduction to Hidden Markov Models. In *IEEE ASSP magazine*, pages: 4-16, 1986.

[93]. I. Redeke. Image & Graphic Reader**.** In *Proc. of 2001 Int. Conf. on Image Processing*, pages: 806-809, 2001.

[94]. C. J. V. Rijsbergen. *Information Retrieval, 2nd ed.* Dept. of Comp. Sci., Univ. of Glasgow, Butterworths, 1979.

[95]. G. X. Ritter and J. N. Wilson. *Handbook of Computer Vision Algorithms in Image Algebra*, CRC Press, 1996.

[96]. I. Rocha and T. Pavlidis. Character recognition without segmentation. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(9): 903-909, 1995.

[97]. H. Sako, M. Seki, N. Furukawa, H. Ikeda, and A. Imaizumi. Form reading based on form-type identification and form-data recognition. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition*, pages: 926-931, 2003.

[98]. H. Samet and A. Soffer. MARCO: Map retrieval by content. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):783–798, 1996.

[99]. H. Samet and A. Soffer. MAGELLAN: Map acquisition of geographic labels by legend analysis. In *Int. Journal on Document Analysis and Recognition*, 1(2):89–101, 1998.

[100]. S. Sarkar and K.L. Boyer. *Computer Perceptual Organization in Computer Vision*. World Scientific, 1994.

[101]. E. Saund, J. Mahoney, D. Fleet and D. Larner. Perceptual Organization as a Foundation for Graphics Recognition. In D. Blostein and Y.-B. Kwon, editors, *Graphics Recognition: Algorithms and Applications*, LNCS 2390: 139-147, Springer, 2002.

[102]. T. Shimamura, B. Zhu, A. Masuda, M. Onuma, T. Sakurada, Y. Kuronuma, and M. Nakagawa. A prototype of an active form system. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition*, pages: 921-925, 2003.

[103]. A. Simon, J.-C. Pret and A.P. Johnson. A fast algorithm for bottom-up document layout analysis. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(3): 273 -277, 1997.

[104]. J. –Q. Song, F. Su, C.–L. Tai and S.-J. Cai. An object-oriented progressive-simplification-based vectorization system for engineering drawings: model, algorithm, and performance**.** In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(8):1048-1060, 2002.

[105]. F.W.M. Stentiford. Automatic feature design for Optical Character Recognition using an evolutionary search procedure. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 7(3): 349-354, 1985.

[106]. M.V. Stückelberg and D. Doermann. Model-based graphics recognition, In A.K. Chhabra and D. Dori, editors, *Graphics Recognition: Recent Advances*, LNCS 1941:121-132, Springer, 2000.

[107]. Y.Y. Tang, S.W. Lee and C.Y. Suen. Automatic document processing: a survey, In *Pattern Recognition*, 29(12):1931-1952, 1996.

[108]. E. Tapia and R. Rojas. Recognition of on-line handwritten mathematical formulas in the E-chalk system. In *Proc. of the 7th Int. Conf. on Document Analysis and Recognition*, pages: 980-984, 2003.

[109]. K. Tombre. Analysis of engineering drawings: state of the art and challenges. In K. Tombre and A. Chhabra, editors, *Graphics Recognition: Algorithms and Systems*, LNCS 1389: 257-264, Springer, 1998.

[110]. K. Tombre. Ten years of research in the analysis of graphics documents: achievements and open problems. In *10th Portuguese Conference on Pattern Recognition, Lisboa, Portugal*, pages: 11-17, 1998.

[111]. O.D. Trier and T. Taxt. Evaluation of binarization methods for document images. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(3): 312-315, 1995.

[112]. S. Truvé. Image interpretation using multi-relational grammars. In *Proc. of Third International Conference on Computer Vision*, pages: 146-155, 1990.

[113]. W.H. Tsai and S.L. Chou. Detection of generalized principal axes in rotationally symmetric shapes. In *Pattern Recognition*, 24(2):95–104, 1991.

[114]. E.R. Tufte. *The visual display of quantitative information*, Cheshire, CT, Graphics Press, 1985.

[115]. B. Twersky, J. Zacks, P. Lee and J. Heiser. Lines, blobs, crosses and arrows: diagrammatic communication with schematic figures. In M. Anderson, P. Cheng and V. Haarslev, editors, *Theory and Application of Diagrams*, LNCS 1889, pages: 221-231, Springer-Verlag, 2000.

[116]. E. Valveny and E. Marti. Hand-drawn symbol recognition in graphic documents using deformable template matching and a Bayesian framework. In *Proc. of 15th Int. Conf. on Pattern Recognition*, Vol.2:239-242, 2000.

[117]. A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. In *IEEE Trans. on Information Theory*, 13(2): 260-269, 1967.

[118]. F.M. Wahl. Deriving features from Hough space for object recognition and configuration estimation. In J.C. Simon, editor, *From Pixels to Features*, pages: 141–152. Elsevier Science Publishers, 1989.

[119]. T. Watanabe. Recognition in maps and geographic documents: features and approach. In A.K. Chhabra and D. Dori, editors, *Graphics Recognition: Recent Advances*, LNCS 1941:39-49, Springer, 2000.

[120]. L. Xu, E. Oja and P. Kultanen. A new curve detection method: Randomized Hough Transform. In *Pattern Recognition Letters*, 11(5): 331-338, 1990.

[121]. N. Yokokura and T. Watanabe. Layout-Based Approach for extracting constructive elements of bar-charts. In K. Tombre and A. Chhabra, editors, *Graphics Recognition: Algorithms and Systems*, LNCS 1389: 163-174, 1997.

[122]. B. Yu and A.K. Jain. A Generic System for Form Dropout. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(11): 1127 –1134, 1996.

[123]. Y.-H. Yu, A. Samal and S.C. Seth. A system for recognizing a large class of engineering drawings**.** In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(8):868-890, 1997.

[124]. R. Zanibbi and D. Blostein. Recognizing mathematical expressions using tree transformation. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(11): 1-13, 2002.

[125]. R. Zanibbi, D. Blostein and J.R. Cordy. A survey of table recognition: models, observations, transformations, and inferences. In *International Journal of Document Analysis and Recognition*, November 2003.

[126]. Y.P. Zhou and C.L. Tan. Hough technique for bar charts detection and recognition in document images, In *Proc. of IEEE Int. Conf. on Image Processing*, vol. 2, pages: 605 -608, 2000.

[127]. Y.P. Zhou and C.L. Tan. Bar charts recognition using Hough based syntactic segmentation. In M. Anderson, P. Cheng and V. Haarslev, editors, *Theory and Application of Diagrams*, LNCS 1889, pages: 494-498, Springer-Verlag, 2000.

[128]. Y.P. Zhou and C. L. Tan. Hough-based model for recognizing bar charts in document images. In P. B. Kantor, editor, *SPIE Document Recognition and Retrieval VIII*, pages: 333-340, 2001.

[129]. Y. P. Zhou and C. L. Tan. Chart analysis and recognition in document images. In *Proc. of 6th Int. Conf. on Document Analysis and Recognition*, pages: 1055 -1058, 2001.