5-2010

Image-Based Math Retrieval Using Handwritten Queries

Li Yu

IMAGE-BASED MATH RETRIEVAL

USING HANDWRITTEN QUERIES

by

Li Yu

A thesis submitted to the faculty of

Rochester Institute of Technology

in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computer Science

Rochester Institute of Technology

May 2010

ROCHESTER INSTITUTE OF TECHNOLOGY

GRADUATE COMMITTEE APPROVAL

of a thesis submitted by

Li Yu

This thesis has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

5/17/2010
Date

Dr. Richard Zanibbi, Chair

5/13/10
Date

Dr. Manjeet Rege, Rreader

5/13/2010
Date

Dr. Roxanne Canosa, Observer

ABSTRACT

IMAGE-BASED MATH RETRIEVAL USING HANDWRITTEN QUERIES

Li Yu

Department of Computer Science

Master of Science

We present a new technique to locate mathematical expressions in document images using images of handwritten queries, without the use of optical character recognition. In our approach, two types of X-Y cutting are used to construct document region indexes that hold potential matches. Candidates are selected based on a set of tolerance parameters and then are ranked using image similarity, with the top 10 matches being returned to the user. The ranking of candidates relies on decreasing sum-of-squares distance between the upper and lower image contours for the candidate region and query image; Dynamic Time Warping is used to compute this distance. We evaluate our system using the maximum region match between a candidate and the region from which a test query is taken. In addition, ten users are invited to participate in the human evaluation, in which they are asked to rank the similarity between each returned candidate and the original query region subjectively.

Two types of queries are used for the experiments and comparisons: the original queries are extracted from the documents directly while their handwritten versions are obtained by asking the ten participants to draw queries following original ones. On average the precision of using original queries is about twice that of handwritten queries. The main contribution of this thesis is the demonstration of the feasibility of this novel query-by-expression method.

ACKNOWLEDGMENTS

I would like to acknowledge Dr. Richard Zanibbi, the supervisor of my master study, who help me a lot not only in the writing of this thesis, but also in my graduate study in RIT. I also wish to thank for the work of Joan Wang, Xi Ouyang, Ling Ouyang, Yuheng Wang, Haohui Yin, Amit Pillay, Hongda Mao, Xiaofeng Fan, Leo Lee and Chaowei Jin who participated in our experiments.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

Methods for fast and effectively locating and accessing text in documents have been studied for many years and a number of successful strategies have been developed [6]. However, for mathematical expressions current text-based techniques are inadequate, as mathematical symbols cannot be easily typed in, and expressions have two dimensional symbol arrangements, rather than strings of characters. In recent years, people have begun work on mathematical information retrieval within digital libraries. Current work focuses on searching through content that is completely marked up based on structure and properties (e.g. using MathML or LaTeX). These methods cannot search through document images, such as in scanned paper collections [7].

We propose a *query-by-expression* method for retrieving mathematical expressions based on their visual and structural features (see Figure 1.1). Pages and queries are segmented by X-Y cut algorithm [4], producing X-Y trees that represent the layout of regions hierarchically. An index of X-Y tree segments is produced, from which candidates are ranked using a visual similarity metric. In this first investigation, we

look at performance for queries taken from directly from the documents, as well as handwritten versions of these queries produced by a group of participants.

The type of 'math spotting' that we are proposing may provide a simple method for people to access mathematical content in digital libraries without the use of optical character recognition (OCR). Our technique also complements current math search algorithms concerned with marked-up structure and semantics, and might be combined with them in the future.

In our experiment, handwritten queries were returning at least one top-10 candidate region covering 43.2% of the original query region, i.e. on average a portion of the specific region sought after was located. In the remaining sections of the thesis we present previous related work in Content-Based Image Retrieval (CBIR), X-Y cutting, and math recognition and retrieval. We then present our indexing and querying mechanisms, followed by our experimental design, results, and a discussion of the results.

Thesis statement: Handwritten queries may be used to retrieve math expressions from document databases using page segmentation and image similarity algorithms, without the use of optical character recognition.

The performance of the method can be evaluated by computing the ratio of the area of overlapped regions to the area of query as well as by asking human users to rank the similarity between returned regions and the query.

1.2 Limitation and Assumptions

Several limitations are acknowledged in our proposed math retrieval method.

1. We only focus on the feasibility of original handwritten math retrieval. It take about 0.4 second to compare one query against one page. Our proposed method

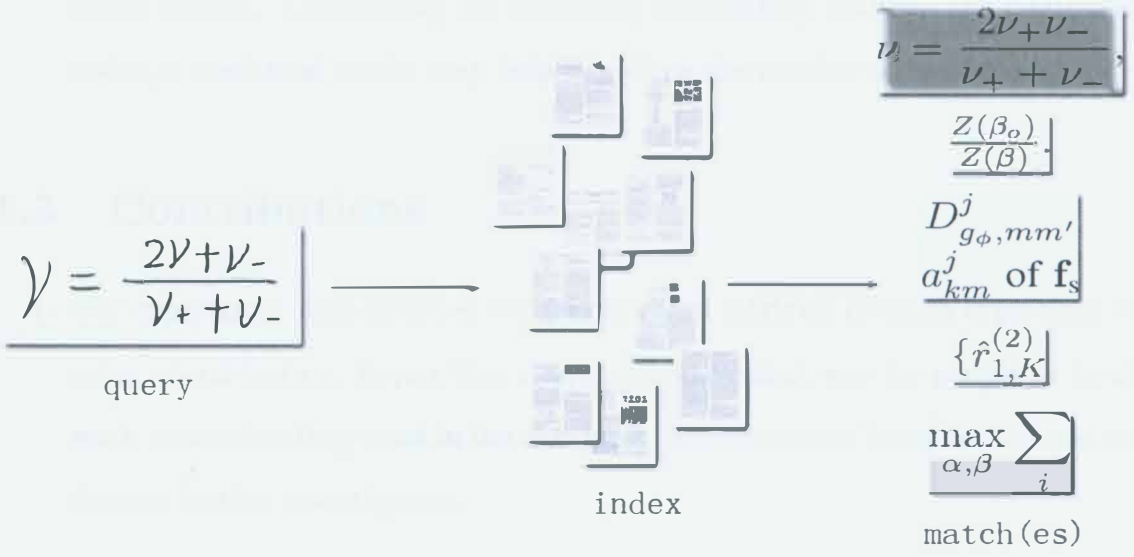


Figure 1.1 Query-by-Expression. Handwritten query images are looked up in a document index, and matched regions in the index are returned.

is not workable in real applications, especially when the document databases are large.

2. Only one similarity comparison method (DTW) is used in our system. Though the accuracy of this method is high, it is more expensive than other methods. The tradeoff between accuracy and speed needs further investigation.

3. The index mechanism used in our system is naive. Since the nodes of X-Y trees are overlapped, we need space that is about five times more than that of original documents to build the index.

4. The accuracy of the handwritten queries still needs to be improved. For those users whose writings are compact (typically the gap between different symbols in such writings is small), the system doesn't work well because of the natural limitation of X-Y technology.

5. Some statistical models need to be investigated to improve the performance of

whole system. Considering the underling relationship between the features of nodes, a statistical model may help to reduce the number of candidates.

1.3 Contributions

1. Our experiment demonstrates the feasibility of retrieval of math expression by using visual-feature. It provides a new and convenient way for people to locate math expression they want in the database. Improvements based on this method deserve further investigation.

2. An overall 90 percent precision is achieved by applying our method to compare the original queries (extracted from the document directly) against the database. The results are gotten from a document database containing more 200 images which are converted randomly from CVPR paper collection 2008.

3. Our method is proven to be workable for spotting handwritten queries. Although the precision is not high compared to that of original queries, 43.2% is a good initial result for handwritten queries.

Chapter 2

Content-based Image Retrieval

2.1 Introduction

With the increase and development of Internet, an increasing number of images are saved in computers, which are usually called image databases. Such images databases are attracting more and more people in different fields. At the same time, a study called image retrieval becomes to be more and more popular. It focuses on getting and accessing target images saved in large databases effectively. The early image retrieval methods tried to find the aim by text matching, which means some kinds of keywords are associated with each image in the database and are used for retrieval. However, considering the huge volume of images in databases, it's infeasible for people to label each image with keywords manually. It's also impossible to link keywords to all images since, in some cases, we can predict all key words. Recent studies pay more attentions to Content-based Image Retrieval (CBIR) method and many commercial products based this method appear. For example, IBM's QBIC image engine and Virage's VIR image engine [1].

2.1.1 Feature extraction

Visual feature plays key roles in CBIR, based on which similarity comparison is applicable. It can be defined as a kind of data that comes from the image and represents the image. For example, color histogram [8] is simple but effective data format that has been widely used for representing the color feature images. Another popularly used feature is called shape feature [9] , which is used to describe the shapes or boundary profiles of images. In addition, texture feature is also widely studied by people in this field. Typically, a CBIR system can retrieve the desired image by any one of the three features discussed above or any combination of them. Figure 2.1 shows the profile [10] of the CBIR problem.

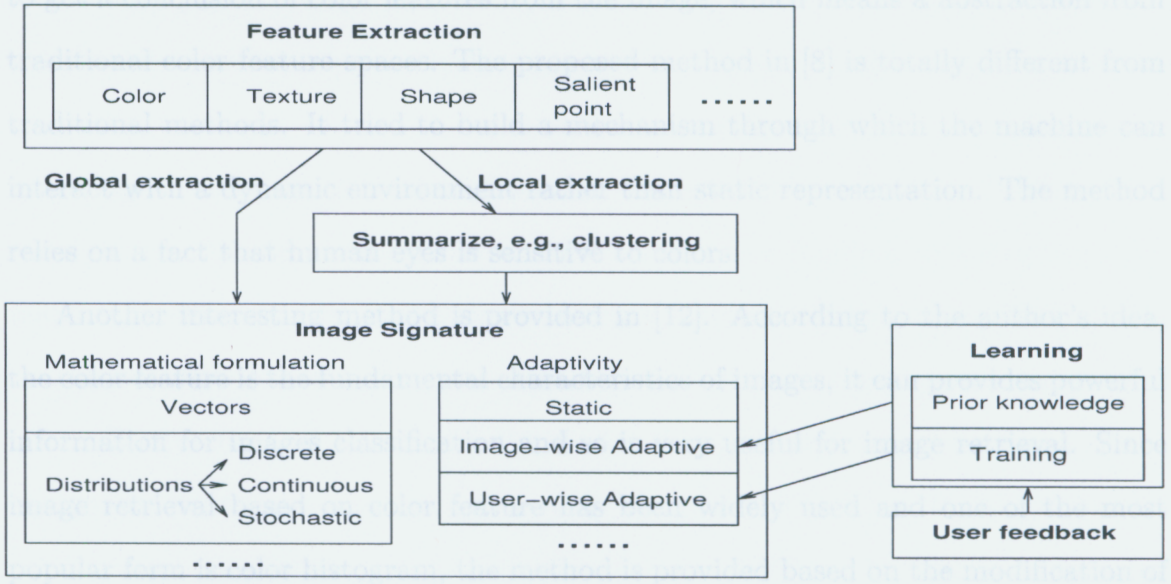


Figure 2.1 An overview of image signature formulation [1]

Feature signatures used for CBIR are usually divided into different categories according to various point of views. Figure 2.1.1 shows different types of image features that are divided based on vectors. Histograms can be regarded as sets of feature vectors and the weights for each element in the vector are used to determine if the sets is a discrete distribution. On the other hand, continuous density usually

provides more accurate description than a discrete distribution with finite vectors. In addition, stochastic model relies on the spatial dependence among local vectors [11], and in some cases, we need such models to represent images. The feature signatures can also be divided into two types: static and dynamic. For the first type, it can be created for all images. While for the later one, the signatures vary according to different classifications, where images are divided into several types first. As shown in figure 2.1, the static feature signatures also contain two types: user-wise and image-wise.

Color features Study of color features was very popular in the field of CBIR, which is focusing on exploration of color spaces recently. The trend in this filed is to get a conclusion of color features from the image, which means a abstraction from traditional color feature spaces. The proposed method in [8] is totally different from traditional methods. It tried to build a mechanism through which the machine can interact with a dynamic environment rather than static representation. The method relies on a fact that human eyes is sensitive to colors.

Another interesting method is provided in [12]. According to the author's idea, the color feature is the fundamental characteristics of images, it can provides powerful information for images classification and so is very useful for image retrieval. Since image retrieval based on color feature has been widely used and one of the most popular form is color histogram, the method is provided based on the modification of color histogram. However, only having the color histogram is not enough to get the desired image in the database, a color space model and a distance metric is adopted by the paper to achieve the goal. The shape of the image is defined as:

$$M_{ij} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^j y^k f(x, y) dx dy$$

where j and k are equal or bigger than zero. This function represents the shape

of pixels associated with color information. The advantage of this method is the improvement of resistence to skew and rotation. The experimental results show the method is effective. It also works well based on global color histograms.

Shape features Another important feature used for CBIR is the shape feature. The word shape is easy to understand but it contains so may invariants. The tendency of using shape feature is transferred from global shape representation to local shape descriptors since the later has a more accurate representation for local objects. In other words, the later one is more helpful for retrieving local objects. One of the popular shape representation methods for local objects is called discrete curve evolution [13] [14] [15], in which the simplification process is used for noise removing.

Another popular shape representation method is called fourier descriptors, which can describe the boundary of images accurately. This method is based on the segmentation of images, which includes many variants and usually more complicate ones mean higher accuracy. In real applications, discrete fourier transform (DFT) [16] is widely used. As one particular form, DFT transfer one function to a frequency domain representation that is also expressed by a function. The input for DFT should be discrete values with limited duration, which is implemented by sampling. DFT is an effective method to transfer the shape of images into functions saved in the computer. It provides a convenient way for the machine to know the shape feature of the image. In addition, the computational cost of DFT can be controlled by using different sampling points.

Texture features Unlike color and shape features, texture feature is more complicated but still widely used for CBIR. The motivation of using texture feature comes from the fact that the retrieval results based on only on color or shape feature usually don't match human perception, including some obviously irrelevant ones. Traditionally, machine view image texture in the frequency domain. Gabor filters and

wavelets [17] are the most widely frequency domain representations. [18] discussed an CBIR based on wavelet transform, in which the image texture is represented as the energy distribution in the frequency domain.

2.1.2 Defining queries

The image retrieval begins with query. So it is important for users to create query images easily and accurately. For text-based retrieval, creating query is quite simple, a typical way is to type keywords associated with desired documents. However for CBIR system, a common way to create query is to draw or select an image. However, in some cases, users are not able to draw the query images and give it to the CBIR system. So several CBIR systems try to provide an interface to gather information that can help users creating query images. For example, IBM's query-by-image-content (QBIC) system provides users with color feature selection that help users to express what they want better. In addition, other kinds of features such as shape or texture are also provided by the system for users to choose. Notice that, in some cases, users are more interested in retrieval of specific objects rather than the entire image and it is also impossible for people the draw a detailed image for each retrieval. For example, to retrieve a picture containing a sunrise, people usually give a query containing only sun without considering the location of sun or its backgrounds. In this case, the system should return all the images containing sunrise. The simplified process of CBIR is shown in Figure 2.2.

Query by visual example Query-by-visual-example (QBVE) [19] is the earliest and most intuitive method to express the desired image and has been studied for many years. The retrieval process is totally based on visual similarity comparison. And the returned items are ranked by similarity. The machine extracts the feature from the query first, and then extract features from images saved in the database.

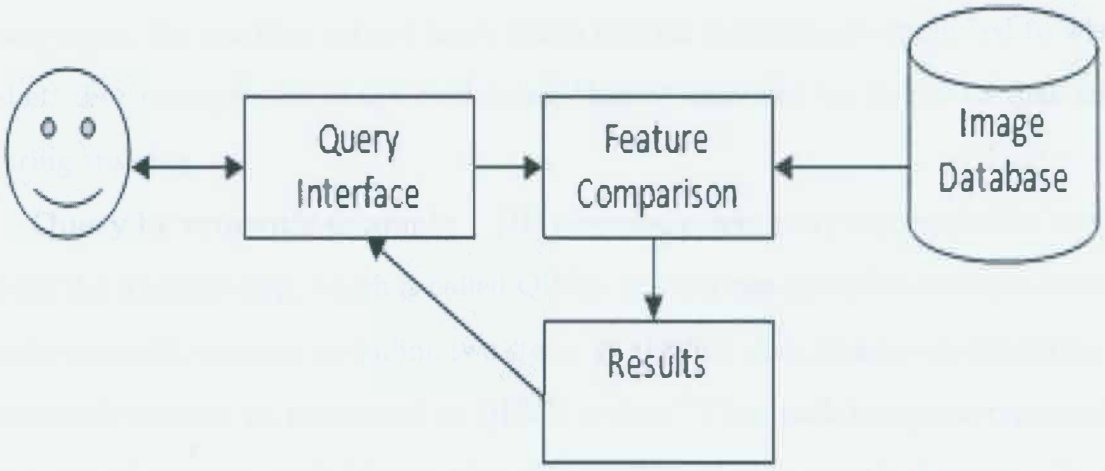


Figure 2.2 Simplified view of CBIR query processing

The query feature is compared against with features of images in the database one by one. Then the ones with highest similarity are returned as results. A number of similarity comparison methods exist, and become more and more complicated in order to get better retrieval performance. The example for QBVE is simple, when a user want to find an image in the database, the system provides him with some query candidate images. Then users only need to choose the image that is most similar to what he wants from the query candidate and the system used this query image for the retrieval.

Semantic retrieval The semantic retrieval [?] [20] becomes more popular recently. Compared to QBVE, semantic retrieval provides a faster way for users to express their desired images. However, this method often fails due the fact that human's perception of similarity is different from machine. It's also call semantic gap. A semantic retrieval system begins with training, in which the computer learns to create a mapping from words to visual features. Then it works following two steps: (1) annotate images with words describing them accurately; (2) search images in the

database using natural language query. The disadvantage of this problem is that in some cases, the machine cannot know which regions in the image are linked to which label. For example, an image containing "horse" may not be linked to that label during training.

Query by semantic example [21] proposed a new query representation method to fill the semantic gap, which is called QBSE. It expresses query-by-example concept in the semantic domain, including two steps. In the first step, images are labeled in an automatic manner as mentioned in QBVE system. Then each images is represented as a set of concept probabilities that are considered as semantic features. In the second step, the classification is performed based on the higher-level semantic space using QBVE method. Actually, QBSE is a combination of QBVE and QBVE and the similarity comparison relies on concept distributions. Experimental results show that in almost all the cases, QBSE has better performance than QBVE, indicating a potential in semantic level retrieval.

2.1.3 Metrics for visual similarity

In a CBIR system, we need methods to compare the similarity and dissimilarity between queries and images. But in some cases, the similarity or dissimilarity is difficult to quantify, which may also vary according to different requirements. For example, adapted from [21], in terms of two pictures, one is a black horse and the other is a red horse. If we only consider the horse, the similarity between the two pictures should be high. But if we are more interested in the color of the horse, the similarity will be lower. So in most cases, it's hard for us find a similarity metric satisfying all kinds of requirements. One widely used metric to evaluate the performance of a CBIR system is called precision and recall. As mentioned above, the precision and recall value is also determined by the different definitions of similarity.

Visual feature based similarity Once we have chosen the features for the image retrieval, the next step is to consider how to use these features for accurate retrieval. The follow shows the key factors involved in the process [22]:

- How to deal with noise
- How to improve the computing efficiency
- How to deal with invariance

At the same time, the methods used in the process can be classified as following [22]:

- Local or global similarity or both
- Linear space or nonlinear space
- Different segmentation strategies
- Fuzzy similarity measurements
- Supervised or unsupervised learning

Figure 2.3 summarizes the features and similarity metrics being studied in recent research. For every feature type, the corresponding mathematical formula and distance is listed on the right hand.

Clustering and classification Typically, image classification is considered as a preprocessing stage. It also followed by some kinds of similarity measurements [23] [24]. Hidden Markov models (HMMs) is a widely used method for image classification. [25] proposed a more complicated 2D-HMMs classification model. In this model, the dependency of the feature vectors is considered in two dimension

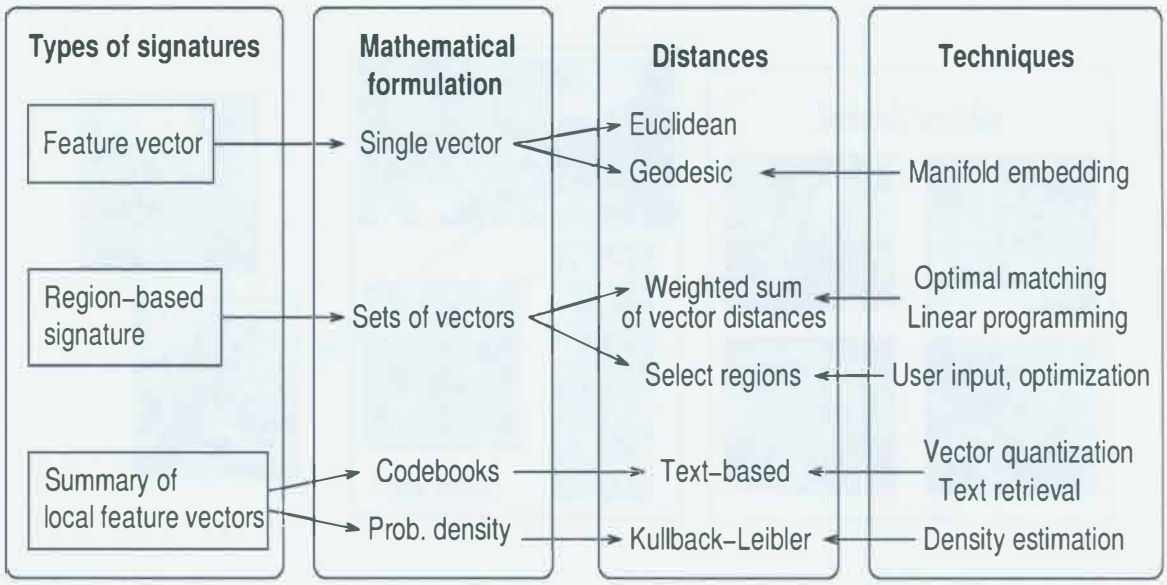


Figure 2.3 Different types of image similariy measure [1]

simultaneously. The parameters for HMM are obtained from the Expectation Maximization (EM) algorithm. Each image is classified based on blocks, the class with maximum posteriori probability selected. The method is based on the assumption that the states might change every block. By taking the relation between current block and its neighbor into account, the method gives more information about context. However, the experimental results only show a slight improvement compared to one dimension HMM model.

Semantic gap Due the different definitions of similarity from the view point of human being and machine, there exists the semantic gap between users and computers. Figure 4 shows an example, a query containing a train resulted in some returned pictures containing bridges since some kinds of shape (eg. arch) similarities exist between trains and bridges. However, in this example, users only want pictures containing trains. So in this case, if the computer compare the query against database using low level feature such as shape feature, the returned items may contain some non expected pictures.

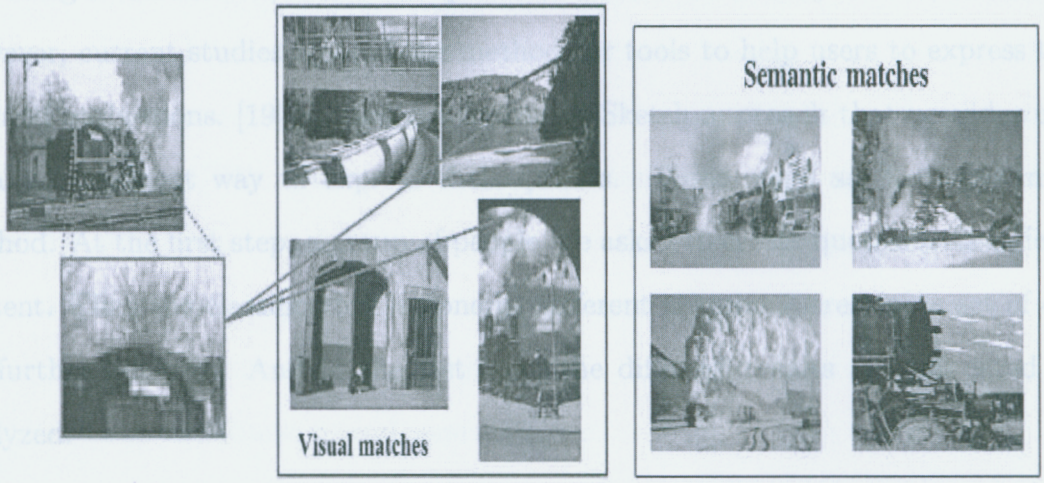


Figure 2.4 Semantic gap [2]

[26] discussed the relationship between low-level color feature and high-level feature. In the experiment, a group of users are asked to tell the similarity of a set of images and semantic clusters are created based on the users' answers. Then this cluster is compared to the classification result using low-level color. To fill the semantic gap, the paper tried to combine low-level color feature with other features to enhance the performance of image retrieval. The proposed method includes three steps. The first step is to cluster the images using both color feature and semantic information from users. The second step is to cluster the images using low level-level color only. Then in the last step, the results based on two different features are compared. It was also inferred from the results that if the color feature is used in specific groups, it can reflect the semantic of them to some extent. So the color feature sometimes can used as initial filter in CBIR.

Evaluation criteria

CBIR is meaningful if only it can be evaluated from the view of human beings. It's difficult to evaluate users' feeling as forms of scores [27]. As mentioned at the

beginning of this section, the value of precision and recall is widely used for evaluation. However, current studies ignored the methods or tools to help users to express their demand evaluations. [19] proposed a Query-By-Sketch approach that provides users with a convenient way to express their queries. Three steps are included in the method. At the first step, a group of people are asked to create queries with different content. Then the classification is done by different criterions, creating a set of data for further analysis. And in the last step, the different results are compared and analyzed.

2.1.4 Indexing and searching

The indexing and searching in CBUR are usually used to improve the speed of the whole system. For example, when comparing a query against images in the database, it is usually not feasible to compare the query with all the images one by one since the image database is large. But in most cases, people only need to compare the query to those with high similarity. In other words, if people can remove images that obviously can not be the target in advance, the efficiency of the whole system can improve significantly. In addition, for some particular image databases, indexing is necessary. In [13], a frequently changed image database is discussed, a method called R-tree is used to index the images dynamically. In addition, even if the distance of the images is not defined as a vector space, a distance function can be defined as well. M-trees [28] has been studied to apply for such situations.

Indexing by hashing

[29] proposed a new indexing method focusing on planar objects. The indexing is based on the combination of projective invariants and voting technique. Experimental results showed the indexing method is effective. The image matching relies on using camera captured documents as queries. Another advantage of this method is that it

is robust to perspective distortion. Basic ideas include three parts: 1. the indexing and retrieval are based on hashing and voting. 2. the has key is invariant-based. 3. local features are used. Actually, the indexing method used in this paper is to widely applied to other CBIR systems. At the first step, all images in the database are registered and the feature descriptors in each image are mapped by a hash function. Then in the retrieval step, the query image is mapped using the same hash function and retrieval results are determined by voting.

Relevance feedback based searching

A strategy used to improve the performance of CBIR is called relevance feedback (RF) mechanism. [30] discussed this mechanism by classifying the image search into two types: target search and category search. For the former type, the goal of retrieval is to find a specific image while for the later one, the goal of the search is to find the a class or genre of images. The relevance feedback method discussed in the paper is based on target search. Four different RF strategies are analyzed, including local neighboring movement (LNM), naive random scan (NRS), neighboring divide-and-conquer (NDC), and global images. Among them, NDC and GDC exploit Voronoi diagrams to reduce the search space to and speed up the movement towards the target. And all algorithms are proved that they can get the target and lead to convergence.

2.2 CBIR for documents

With the increasing number of documents being transferred into computers, people found that it's not convenience for them to get access to information in these documents since most of these are stored as images. This is because it's hard for people to transfer the content of these documents words by words and in most cases, they do it by using some devices such as camera and scanner. To solve this problem, many

methods have been proposed. One of the most popular way is called optical character recognition (OCR), which means the content in document images is converted to some format that can be understood by the machine first. However, in real applications, a large amount of content in documents such as non-text and rare symbols can not be transferred into electronic formant completely. In these cases, the recognition methods don't work well. So people need to find other ways to get and index useful information in the document images, which means much more complicate tasks when compared to tasks based on electronic document format. In our study, in order to give a complete view of methods used in document retrieval. We first give an overview of traditional text retrieval methods based on electronic document and then we will discuss about the image-based methods [31]. Notice that although many studies have been done in this field, there are still more challenging tasks related such as extracting key information, auto indexing, etc.

2.2.1 Text documents retrieval

Accessing information in text document is a very old topic in the filed of information retrieval, which focuses on the analysis of electronic sources such as news papers and technic articles. All of these works rely on an assumption that the content of documents are represented by electronic format. According to the different sources, the methods in this field can be divided into two types: retrieval of information in traditional documents and processing information based on converted text. Although these studies have noting to do with document images, they give us a lot of enlightenment due to the internal relationship between text retrieval and image retrieval.

Traditional indexing and retrieval Usually, documents can have two kinds of components according to their structures: structured and unstructured [32]. For instance, delineates in emails can help people to distinguish different components in

the document. These components include author's names, titles, etc. If such identifers can be found by the machine to refer to specific components in the documents, it much easies for people to access information than analyzing the content of the entire document, which means a more complicate and challenging problem.

The basic method for text indexing and retrieval relies on some kinds of statistic features such as the text corpus and frequency of key words. It's also called document content characterization. Many methods have been tried in this field, [31] [33] proposed a method in which common "stop words" that don't have too much useful or interesting information are filtered out from the documents at the first step, and then in the second step, the document information is represented by a vector storing the frequencies of meaningful terms in that document. For example, a vector T (w_1 f_1, w_2 f_2, ... w_n f_i) give a frequency representation of meaningful terms in a document, where w_i refers to the key words or meaningful symbols involved in the document and f_i means its corresponding frequencies. In addition, suffixes are removed and both low-frequency and high-frequency words are replaced with some equivalences according to a set of rules. Once a document is indexed, the feature vector is created and saved for further retrieval. If people want to retrieve some specific document in the collection, they only need to compare the vector of the query against all the vectors generated from the documents in the collection. The one with minimum vector distance will be returned as the result. Actually, this basic method provides a profile for all kinds of information retrieval problems such as image retrieval rather than mere text retrieval.

Converted text process Another popular topic related to text retrieval is processing of converted text. The main tasks involved in this topic is noisy filtering. The problems are derived from noise included in the electronic document format converted from document image using OCR. It is closely related to our image retrieval

problem since it can be considered to be an recognition improvement stage. In order to encourage studies in this filed, the Test Retrieval Conference provides a opportunity for researches to evaluate their methods and compete with others and all the methods should be based on the converted documents. The confusion contest in a conference started since 1996. The test data in the contest is obtained from Federal Registry, which is the result of recognition of some kinds of printed and scanned documents. The groups taking part in the contest are given the original documents as well as the converted ones. They are also provided by the same set of queries. The performance evaluation relies on precision and recall values. The performance of programmes are evaluated on both corrupted and uncorrupted data. Typically, there is a significant drop in retrieval precision between results from corrupted data and uncorrupted data.

[34] proposed a method based on the modeling of errors in the output of OCR. A set of symbols and page models that may lead to significant accuracy demote was listed. Most of them are typical errors that are usually occurring during scanning and recognition. The advantage of the proposed method is that a set of parameters can be adjusted to simulate a wide variety of conditions with relatively low cost. Through the study, a direct relationship between recognition accuracy and retrieval precision is revealed. For example, they found that if the OCR accuracy is 90% or above, most retrieval techniques work well and while the accuracy is between 70% and 90%, only about half of methods work well and when the accuracy of OCR is below 70%, almost all the information retrieval methods fail due to errors.

Comparison As discussed above, the text documents retrieval and image documents retrieval are similar if we consider them in a same profile: feature extraction and similarity comparison. The major difference between text documents and image documents retrieval are the features used for the retrieval. For text documents, features are relatively easy to obtain since the documents are represented as electronic

format the computer can understand. For example, for structured text documents, we can get useful information by identifying specific identifiers while for unstructured text documents we can get important indexing information by computing the statistical values as discussed above. However for image documents, the features can be used are much different since they are gotten from the image pixels. The recognition errors shown in section 2.2 play the same role as noises in image documents features, which may have significant influence on the accuracy of the documents retrieval.

2.2.2 Image documents retrieval

From the information retrieval, similarly, in order to perform retrieval on document images, we need a effective way to represent the features of the document content [32]. A number of studies have focused on various problems such as proper nouns identification, image abstraction and key words spotting. Most of these methods works well when facing with poor image quality, in which case most OCR methods have very poor performance.

Text feature extraction DeSilva and Hull [35] have studied the problem of finding proper nouns in document images. Such proper nouns include name of people, places and objects. The key part of the work is the algorithm used to extract the features from images. The document image is segmented into words level and the proper nouns are removed by exploring specific features of the word image. Based on a large data set, the paper tested seven features including capitalization, word length, word location, etc. Several interesting properties are discovred by the the study. For example, over 95% of proper nouns are capitalized and over 35% of capitalized words are proper nouns. In addition, less than 10% of proper nouns are longer than the average length of all words and more than 85% of capitalized words are proper nouns if they are followed by a single letter. Although the method is only about feature

extraction, it provides us some interesting investigations. At least, we can know that features that are difficult to use for recognition can be extracted from the image-based expressions. Furthermore, by using feature from the image content, the processing of recognition actually can be avoided.

Word spotting Word spotting is another problem involved in image documents retrieval. A lot of work has been done on spotting key words using visual features. Key words are considered to be useful for document image indexing since usually they can be accessed on image level, which means additional recognition can be avoided. The features used for key words spotting are usually based on shape because it can keep constant while some other features such as font size, style and font vary. Several studies have been addressed on this field with different advantages and disadvantages. [36] proposed a segmentation free method for word spotting. In the first step, candidate lines of text are selected using shape information. The top-down and bottom-up profiles of each word are computed and used as parameters for a Hidden Markov Model (HMM). The key word HMM is created using a set of characters and the Viterbi decoding is used to recognize the key words and corresponding variations in the line. Experimental results show a detection rates of over 95% on a restricted range.

Automatic abstracting of images Another interesting topic related to this filed is about automatic abstraction, which is also based on image. Although the problem has been studied for many years, none of them has been done on document images. Actually, it is not like word spotting problem because automatic abstracting is more complicated. [37] proposed a system to select a set of portions from an document image for summary. The system relies on an existing abstracting technology working on text documents. However, the traditional method usually usually removes stop words first, which is not applicable for images. So in order to applied

it to images, the stop word should be recognized first. Another problem is that we need a new method to compute word equivalences in images. It seems that all these steps are difficult to complete without word recognition. It this case, the method extracted words and segmented them into different categories. A set of statistical values is used to identify the locations of words in the document. The work gives a new and interesting method for summarization of image-content. In sum, automatic abstracting of images is a challenging work in image document retrieval. The method discussed above suffers from many limitations and cannot satisfy the requirements of applications. The problem is caused by the unknown relationship between the meaning of words and their statistical characterization. It seems that it's really difficult to determine if the words are the ones we want only by considering their length and their positions in the document.

2.2.3 Index using documents structure

Document structure plays important role in document images retrieval and a lot of work has been done based on the output of a document analysis system. The large amount of studies is partially due to the variety of forms of existing structured documents. Typically, there are two kinds of structures used to index a heterogeneous document: based on physical structure or based on logical structure [32]. And here, I only give an example of using physical structure and later compare segmentation-based method to segmentation-free methods.

Segmentation-based methods The method for indexing document images based on physical structure is proposed by [38]. Because of lacking of methods for non-text retrieval, the author builded a relational database and provided a layout editor to help users creating queries. The idea is derived from the fact that personnel filing systems are often achieved using layout information [39] [40]. The layout retrieval

prototype was developed under MS Windows and several different search techniques were evaluated. Typically, the performance of method for text search is lower than that of layout search, but in the paper, the strategy is a combination of the two methods, which increases the performance of layout retrieval obviously.

Texture-based methods [41] proposed a system using one type of texture feature called Moravec operator (one popular method to get texture feature by computing the variations in every pixel's neighbors). The basic idea is that queries need to be associated with layout information to improve the retrieval accuracy. The real application, the texture feature is actually a broad concept instead of the Moravec operator discussed above. Other common texture features used for document retrieval include font size, font quality and distribution of diagrams, etc.

2.2.4 Applications

For the new applications in this field, the most popular ones include document image matching, indexing image captions and document image compression. In [42], Hull proposed a method for matching documents having same character content but may having been reformatted. While in [31], Srihari tries to use the association between images and their captions to try to do recognition and content-based retrieval. These two applications combine different image documents retrieval technologies discussed in previous sections and indicate the tendency in image documents retrieval.

Document image matching In [42], a method to organize a database of document images is studied. The method is workable for the situation that the location of matching documents have been reshaped so that they look totally different. Each document image is expressed by a set of descriptors invariant to distortion, scaling, and rotation. Individual descriptor only extracts local information and the entire set of descriptors give a redundant description of its content. The results

show that the best performance is obtained when all the descriptors are compared to the documents in the database, which means the top ranking is always correct. In addition, the distance between first and second rankings can be used to set a constant threshold.

The highlight of this paper is the indexing strategy, which use hashing to speed up the matching process. This approach has been used not only in that paper but also used to solve a series of similar problems. The algorithm contains two basic steps. The first step is used for setting up a hash table that can save all descriptors extracted from the document, which is implemented by locating all the regions that contain text in the document. Then each descriptor is input to a hashing function. A list of identifiers is builded for each entry. When a query comes, it will be represented by a set of descriptors first, then this set of descriptors is converted by the same hash function. The final decision is determined by sone kind of majority voting mechanism.

Indexing image captions Researchers tried to establish a relationship between the content of title and body as well. In other words, it might be possible to retrieve relevant images using information from the title of the picture. [43] studied the the relationship between an image and its caption and try to use it for retrieval work. The work is based on natural language processing and is associated with domain knowledge. The system can recognize images effectively by using gender differences in the language and spatial relationships in the picture content. In this work, both visual feature and text features are used.

2.3 Summary

Visual features such color, shape and texture have proven to be very useful for retrieving image in image databases. The methods used in CBIR system are workable

when applied to document images as well. Math expressions as one type of content in the document images share similar features with other kinds of content such as text, so the strategies used in CBIR system might be also applicable for potting math expressions in document databases. However, unlike common text lines or keywords, the math expressions definitely have different styles of representations, which means if we should exploit some methods or at least some new applications based on existing methods. Considering the unique features of math expression, several segmentation technology, especially some kinds of "precise" technologies, might be very helpful for the later retrieval task since, in most cases, math expressions have totally different structures compared to the common text content. With respect to the retrieval stage, an image comparison mechanism is indispensable. And some kinds of pixel-level similarity comparison algorithm might be useful for comparing math content in the document images.

Chapter 3

Methodology

The steps included in our project can be divided into four parts. The first one is preprocessing step that is also called "index building". In this step, a simple but effective indexing algorithm is implemented to build a new database for math retrieval. The second one is segmentation in which the query image is represented as a X-Y tree for comparison. The third step includes can be seen as candidates selection, a set of parameters are used to filter noise and choose nodes with higher possibility. The last step uses DTW algorithm to compare the similarity between the query and candidates.

3.1 X-Y cut

The use of X-Y cut technology discussed in this thesis is based on the assumption that the page is skew free (i.e.the page is not rotated). Also, we are querying .pdf files in our experiments, which have no skew. To construct the index and query trees, both recursive X-Y cut and standard X-Y are used in the system. By applying recursive X-Y cut to the pages, each page is decomposed into a set of rectangular blocks. At

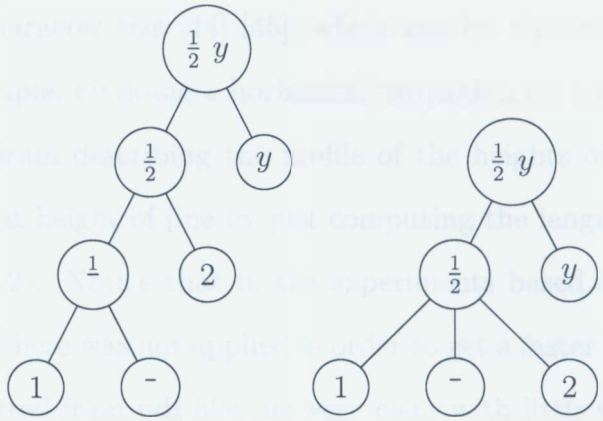


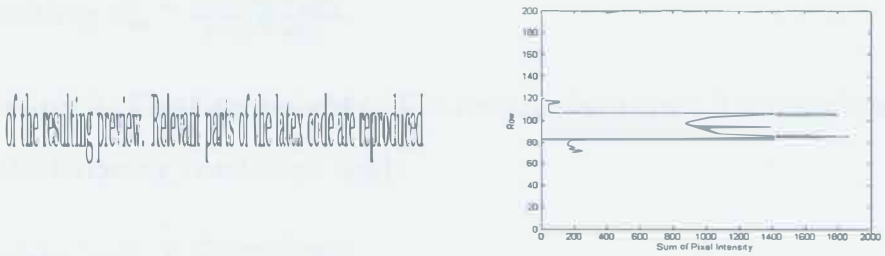
Figure 3.1 Recursive X-Y tree [3] and standard X-Y tree [4] for the expression '$\frac{1}{2}$ y.' Cutting alternates in the vertical and horizontal directions for standard X-Y trees, while recursive X-Y cutting splits the image at the largest horizontal or vertical projection gap at each node.

each step, both vertical and horizontal pixel-projection profiles are calculated. Then a cut is performed at the maximum blank gap and the process is repeated recursively until no gaps exist in either direction. After the cut, each page can be represented as a binary X-Y tree. The root of tree represents the whole page while the leaves represents the minimum components that cannot be divided.

The standard X-Y cut is used as noise filter. By cutting the region vertically and horizontally once each, the numbers of components of both direction can be used to reduce the number of candidates.

Figure 3.1 shows two types of X-Y tree for the expression $\frac{1}{2}$ y. The left one is obtained by recursive X-Y cut while the right one is from standard X-Y cut. As described above, the difference between the two X-Y algorithm is the order of cutting directions. For the recursive one, the cutting direction at each step is determined by the maximum gap of the both direction and the larger one will be cut. For the standard X-Y cut, it cuts the image in vertical and horizontal direction alternatively. In addition, threshold values are used to control the cutting. They are determined

by the dominant character size [44] [45], which can be obtained using a histogram technique. For example, by doing a horizontal projection on a line in the document, we can get a histogram describing the profile of the heights of that line. Then we can get the dominant height of line by just computing the length of the peaks in the histogram (figure 3.2). Notice that in the experiments based on CVPR paper, the thresholds discussed here was not applied in order to get a faster training speed. Since the jpg files transferred from pdf files are very clear with little noise, the application of thresholds makes almost no difference on the retrieval rate. However, the effects of thresholds are obvious when they were applied to documents with noise. For example, we applied they to the UW3 data set to get better retrieval accuracy.



(a) One line in the document (b) Horizontal projection

Figure 3.2 Determine dominant character size

In the implementation, the thresholds for each page are calculated separately. Rectangles that are narrower than the thresholds will be ignored in the cutting. This avoids poor cuts cause by noise, and symbols that overlap horizontally and vertically. As can be seen from figure 3.2, the width of peaks can be calculated by setting a threshold to filter smaller values in the horizontal projection histogram. In our implementation, the thresholds for each page are calculated by the method described above separately. Those rectangles whose size smaller than the thresholds will be ignored in the cutting. So the "bad" cutting (caused by noise) can be avoided.

In addition, the thresholds can control the way in which regions are cut and prevent them from "bad" division. For each page, thresholds are calculated based on the most frequent character heights C_h and blank widths C_w in that page. First, by horizontal cutting, we get the a histogram (h_1,h_2,...h_n) representing the heights of lines. C_h = Mode(h_1,h_2,...h_n). Second, by vertical cutting, we get a histogram (w_1,w_2,...w_n) indicating widths of space regions in a line. C_w = Mode(w_1,w_2,...w_n). At each cutting, the thresholds C_h^n and C_w^n are scaled linearly based on the current region's height and width [45]. The process of getting height threshold is described in Algorithm1.

Definition 1: Cutting widths. The rectangular region R is a vertical cutting region if the following conditions hold:

- R.width $\geq C_w * \frac{CurrentWidth}{PageWidth}$.

Definition 2: Cutting heights. The rectangular region R is a horizontal cutting region if the following conditions hold:

- R.height $\geq C_h * \frac{CurrentHeight}{PageHeight}$.

3.2 Indexing

Our index used to store candidate regions for query matches is constructed using information obtained from the subtree of the page X-Y tree corresponding to the region. During search, regions located within bins matching related X-Y tree properties of the query or lying within given tolerances within the index are considered as candidates. Returned candidates are then ranked using an image matching algorithm (Dynamic Time Warping). Figure 3.3 gives the process of the index and search strategy which is described in subsequent, and related details are summarized in the remainder of this section.

Algorithm 1 SetThreshold (image:BwImage)

Set Hproj to be an integer histogram

Set Hpeakwidth to be an an integer histogram

$Hproj \leftarrow HorizontalProjection(image, PageRect)$

$PeakWidth \leftarrow 0$

for $i \leftarrow hproj.begin$ to $hproj.end$ **do**

  if $i == 0$ and $i + 1! = 0$ **then**

    $PeakWidth \leftarrow 0$

  else if $i! = 0$ and $i + 1 == 0$ **then**

    Add $PeakWidth$ to Hpeakwidth

  end if

  $PeakWidth \leftarrow PeakWidth + 1$

end for

$maxNum \leftarrow$ The maximum value in Hpeakwidth

set countArray to be an integer Array

$countArray.size \leftarrow maxNum$

for $i \leftarrow Hpeakwidth.begin$ to $Hpeakwidth.end$ **do**

  $countArray[i] \leftarrow countArray[i] + 1$

end for

$heightThreshold \leftarrow$ the index of maximum in countArray

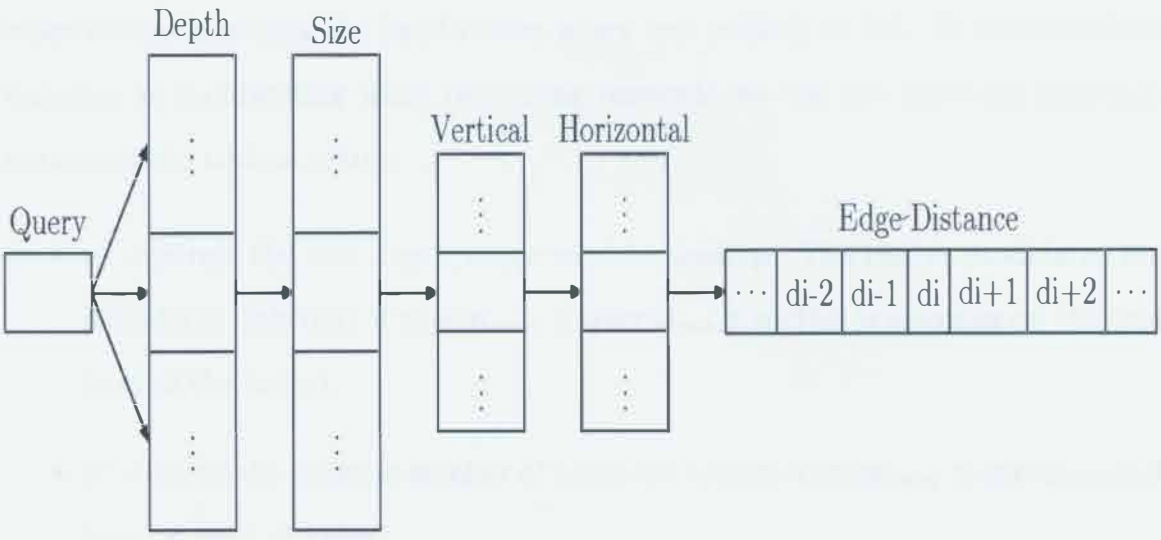


Figure 3.3 Candidate region index. Candidates are located based on their X-Y tree depth, size (number of nodes), number of vertical and horizontal elements in a top-level X-Y cut, and distance from primitive regions to the image boundary.

All the pages in the data set are divided into small regions and saved as the new images, each one of which represents a node in the binary X-Y tree. The nodes chosen to save are controlled by the size and depths of the node. The index only contains regions whose recursive X-Y tree contains less than 100 nodes and have the depth value more than 3 (this avoid including single characters in the index). The first value is used since for a binary tree with size n, the number of its leaves is $[\frac{n-1}{2}] + 1$. A node in X-Y tree with size 100 means 50 connected components (smallest blocks after X-Y cut), which is big enough for common math expressions.

3.3 Searching

We use the root of the query recursive X-Y cut tree to compare a query to regions in the document index. Obviously, it's computationally unacceptable to compare the the root with all the nodes of the page tree, and with handwritten queries looking only at

index entries matching the handwritten query tree is likely to fail. To accommodate variation in handwriting while restricting research, we use the following tolerance parameters to control search:

- α: controls the tree depth range used for lookup. The region candidates are considered only only if $region_{depth} \in query_{depth} \pm \alpha$ (this is operates on the first level of the index).

- β: controls the range in number of nodes for lookup: $region_{nsize} \in query_{nsize} \pm \beta$ (second level of index).

- γ: number of leaves of the X-Y tree obtained by doing a standard X-Y in vertical direction once) and horizontal components number (similar definition). These control lookup in the third and fourth levels of the index.

- δ: controls the range of distance from the edge of the image bounding box to the vertical components. This is illustrated in Figure 3.4; using the vertical projection profile on the page region, we can get the components of each region. (each component is included in a dotted box). Then for all the components $\{I_1, I_2, ...I_n\}$, the distance from both the top edge and bottom edge are calculated and saved in two sets: $top\{I_1, I_2, ...I_n\}$ and $bottom\{I_1, I_2, ...I_m\}$. $\delta = \frac{min(max(top),max(bottom))}{image-height}$ and $region_{distance} \in query_{distance} \pm \delta$. Entries in the last level in the index are sorted by this vertical distance value, and so finding the interval containing regions within the δ tolerance can be performed in $O(log_n)$ time using binary search, following by scanning in either direction to find candidates within the tolerated distance value.

(a) word

$$\sum_{k=1}^{K} \pi(k)\mathbf{p}_k(x)$$

(b) math

Figure 3.4 Vertical Components

3.4 Dynamic Time Warping

All candidates returned by the index lookup are then ranked by image similarity. A number of image matching algorithms might be used in this step, those most widely used including *XOR* [46], *SSD* [46] and *EDM* [47].

- XOR is the most basic matching technique implemented by taking a simple XOR of two images. The white pixels in the XOR image indicates mismatches between the two images.

- SSD is short for Sum of Squared Differences. In computing SSD, one image that has smaller size is used as the template and the upper left pixel of the template image is then aligned with a pixel on the image to be compared.

- EDM is used improve the XOR matching algorithm by weighting the white

pixels in the XOR image according to their distance from black pixels.

We tested several algorithms empirically on a sample of the University of Washington III Technical Document Database [48] and found the DTW (Dynamic Time Warping) [5] method have higher accuracy than other methods. DTW has been widely used to compare time axes in a non-linear manner. When it has been used for word spotting, the time domain was transferred to the feature domain that was represented by the feature vector. So in this case, each element in the feature vector refers to one point in the time axis. The value for each point in the time axis is replaced by some kind of visual information for each column in the image. So the dissimilarity between two images can be obtained by DTW to compute the distance between two horizontal axes. Notice that for each point in the horizontal axis, there may exist multiple feature values.

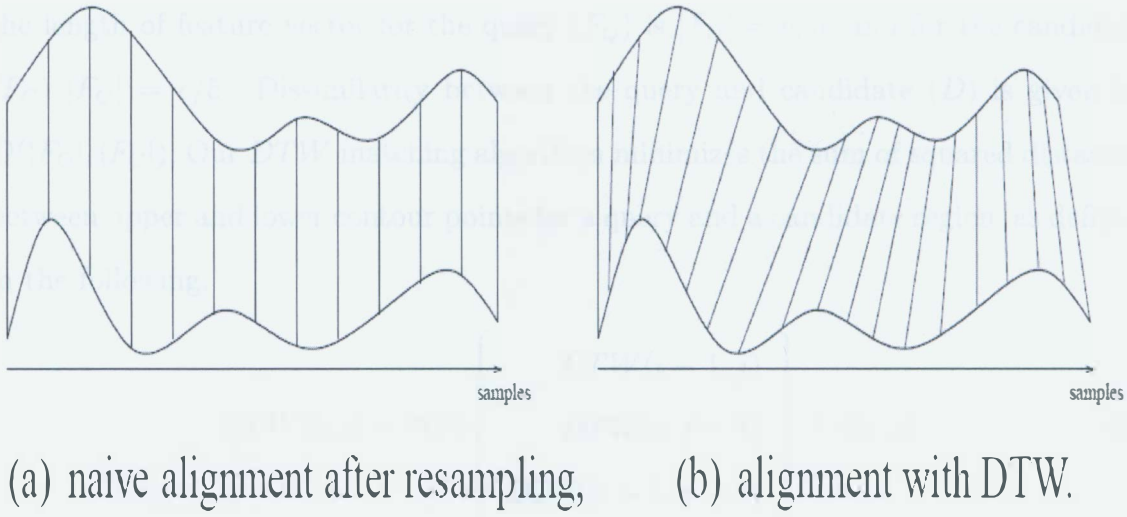


Figure 3.5 Different alignments of two similar time series [5].

Figure 3.5 shows the different alignments of two similar time series. As shown in the left of the figure, a simple method to compute a matching distance between two time series is to compare sample-by-sample. While in the right of the figure, the

Dynamic Time Warping Algorithm computes the distance between two time axes in a non-linear manner, which results in a wrapped space. The effects of such wrapping are shown in the figure in an very intuitive way.

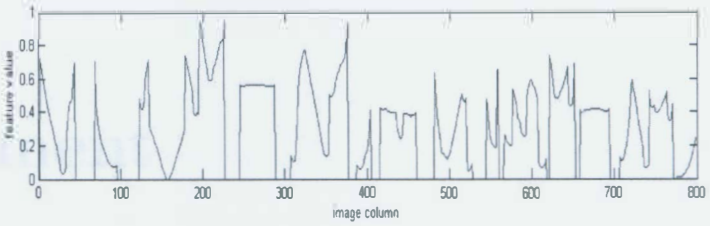
In our experiments, candidate regions are normalized to the same height as the query by computing the ratio of the height of the candidate to the height of the query. A feature vector that consists of two features every five columns of the image is then extracted from each query (I_Q) and candidate (I_C). Figure 3.6 shows the pixel projection profiles from both top and bottom of the image. In order to reduce the time of running DTW algorithm, for every five columns of the images, the average distances from the top edge and bottom edge to the nearest black pixels are calculated respectively. For images whose widths can not be evenly divide by 5, the average value of remainder columns is computed and put into the last position in the feature vector.

For example, if the width of the query is q pixels and the candidate c pixels, the length of feature vector for the query (F_Q) is $|F_Q| = q/5$, and for the candidate (F_C) $|F_C| = c/5$. Dissimilarity between the query and candidate (D) is given by $D(|F_Q|, |F_C|)$, Our DTW matching algorithm minimizes the sum of squared distances between upper and lower contour points for a query and a candidate region, as defined in the following.

$$DTW(i,j) = min \left\{ \begin{array}{c} DTW(i-1,j) \\ DTW(i,j-1) \\ DTW(i-1,j-1) \end{array} \right\} + d(i,j) \tag{2}$$

$$d(i,j) = \sum_{k=1}^{2} (f_k(F_Q,i) - f_k(F_C,j))^2 \tag{3}$$

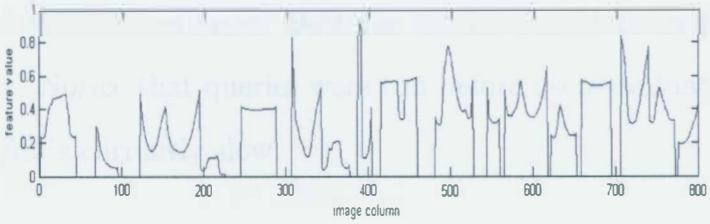
where $i \in [1, |F_Q|]$ and $j \in [1, |F_C|]$, f_k represents the upper and lower contour profiles.



(a) Top-half Pixel Projection

$$A(\hat{W}_A - V_A) + N(W_N - V_N)$$

(b) Query



(c) Bottom-half Pixel Projection

Figure 3.6 Example of upper and lower contour profiles used for image matching (via Dynamic Time Warping).

Chapter 4

Experiment

The experiments are based on 400 documents randomly selected from CVPR 2008 collection containing 1688 pages. Since the original papers only have "pdf" format, the files are transferred to "jpg" format using 300 DPI resolution. Two kinds of query images are used for the experiments: original query and handwritten query where the former one is used as control. In addition, both the machine and human users participated in the experiment. Both the machine and human users participated in the experiment. Notice that queries were run before users evaluated them, in part because the system is currently slow.

4.1 Data set

The dataset used for this experiment is extracted from the CVPR 2008 paper collection which contains a total number of 1688 document pages. All the pages are converted from PDF to JPG first, and then 400 pages are selected randomly from them as the dataset. In the dataset, 200 pages are used for training and the other half for testing.

4.2 Query set

The queries used in the experiment consist of two types: original images from the data set, and handwritten versions of these images. Our procedure for defining our set of original image queries is given in the following:

- According to the layout structure of math expressions, we divide them into three types: vertical, horizontal, and combined type. Vertical includes expressions that only can be cut vertically at the first time ($x+y=z$). Horizontal is defined in a similar way ($\frac{x+y}{2}$). Combined expressions can be cut in both directions on the first call ($\sum^{i-m/2} c_j$).
- Exactly one query is selected from each page containing math in the data set. If a given page contains more than one type of query, try to exact one that has a type with smaller counting number.
- Then randomly sample from these candidates to produce the original image query set.

For training and testing we used indexes built from 200 document pages (roughly the number of pages in an average book). 68 pages were found to contain math in the training set, and 61 pages of the testing set pages contained math. We follow the procedure above, and then for each set, 20 queries are selected randomly. So the total number of original queries is 20 * 2 = 40.

For handwritten queries, we had 10 participants draw the 40 queries on paper, with the original query to the left of the drawing area for each expression. Participants were not told which expressions were for training or testing. In the end, 200 handwritten queries were produced for both the training and testing sets. (Appendix A)

4.3 Training and testing

Five metrics are used to evaluate the performance of the system. Each query is defined using an expression taken from images in the training or testing data sets. We compare results for querying using this original image (as a control) to handwritten versions of the expressions created by the participants in our experiments. To characterize query execution, we recorded the number of pages on which candidates were located (q_p), the number of candidates per page containing candidates (q_c) and the time for a query (q_t). The quality of returned matches were evaluated using the following two (conservative) matching metrics, for each of the top-1, top-5 and top-10 results:

1. Maximum percentage area of match: $\frac{|Region_q \bigcap Region_t|}{|Region_q|}$, where $Region_q$ is the query bounding box and $Region_t$ is the bounding box of a returned candidate region (A_{match})

2. The percentage of queries returning the document page of the original query (P_{match})

As we use only the top ten candidate regions in our user evaluation, the training in this experiment is used to find a set of parameters that maximize the average match area size (i.e. average A_{match}) for our training sample of handwritten queries. A training sample of 200 CVPR pages are used, and the query set contains 20 queries handwritten by 10 users (200 queries in total). We perform a grid search, using the following parameter ranges: $\alpha = \{0, 2, 4, 6, 8\}$, $\beta = \{0, 2, 4, 8, 16\}$, and $\delta = \{0.05, 0.1, 0.15, 0.2\}$. The parameter for filtering based on the number of horizontal and vertical components in a top-level X-Y cut (vertical cut first) is fixed in our experiment ($\gamma = 2$). For our data, we found that the region match of the handwritten queries to the original images was maximized at $\alpha = 4, \beta = 16$, and $\delta = \{0.2\}$ (see Table 4.2).

The performance of the system was then assessed using a separate sample of 200 pages from CVPR and 20 new query images from this set. Every participants that provided training queries also provided test queries (10 participants producing 200 queries in total). Each participant ranked matches for their own handwritten queries, as well as results for the original query images (20 original images and 20 handwritten queries were evaluated by each participant). All participants were computer science graduate students at our institution. Handwritten queries were run off-line before participants were brought in to evaluate matches.

A portion of the user interface participants used to rank matches is shown in Figure 4.1. Participants evaluated the similarity of queries and returned regions using a 5-point scale:

- Rank1: No match.
- Rank2: Less than half the query is matched.
- Rank3: Roughly half the query is matched.
- Rank4: More than half the query is matched.
- Rank5: The query is completely matched.

4.4 Results

Figure 4.2 shows changes in the average maximum region match A_{match} for the location of the image used to define a query obtained during training. Based on these values, $\alpha = 4, \beta = 16$ and $\delta = 0.2$ were selected for the testing phase. Performance did not change for values of α larger than 4 (defining the range of depths with which queries will be matched). Increases in β and delta on the other hand always increase

the average maximum region match, and so we used the largest values for this preliminary experiment. Queries were run off-line before participants performed the user evaluation, so user evaluations were not affected by query time (q_t) in our experiment.

For the chosen tolerance parameters, metrics for quality of match and query performance are shown in Tables 4.1, 4.2 and 4.3. For the top-10 candidates returned, we have a mean query area match per participant of 43.3%, meaning each query on average returned a candidate covering 43.3% of the page region used to define the query (the original image). Similarly, on average a returned candidate was from the page from which the query was taken 63.2% of the time on average. Note that these are very conservative metrics, as they do not include candidates that are similar or even identical to the query if the candidate appears on a page different than the query.

Table 4.1 Training Match Quality ($\alpha = 4$, $\beta = 16$, $\delta = 0.2$). 10 participants, 20 queries each

Metric	Top	μ(%)	σ(%)
P_{match}	1	37.5	12.4
	5	52.5	13.6
	10	60.5	14.5
A_{match}	1	24.0	12.6
	5	36.0	13.1
	10	40.9	14.4

Table 4.2 Training Query Performance ($\alpha = 4, \beta = 16, \delta = 0.2$), 10 participants, 20 queries each, q_c:num of candidates for each page, q_p:num of compared pages for each query, q_t:operation time for each query

δ	$\mu(q_c)$	$\sigma(q_c)$	$\mu(q_p)$	$\sigma(q_p)$	$\mu(q_t)$	$\sigma(q_t)$
0.05	12	6.1	115	17.1	13s	7.7
0.1	21	9.9	144	21.1	22s	13.8
0.15	32	12.1	157	17.3	34s	18.1
0.2	45	15.9	168	17.8	48s	23.3

For testing we present both the user evaluations for the top 10 candidates along

with the maximal query region match A_{match} for each participant's queries, and for the images taken from the test set to define the queries used.

We can see that the match metric for the original queries is very high, and all matches made are returned as the top result, with an average A_{match} of 90%. Participant ratings are represented using the maximum rating (1-5) returned for each query. For rating their own queries, the average best rating in the top 10 candidates was 3.15, where a rating of 3 indicates that roughly half the query is matched. Not surprisingly, participants rated matches for the original images higher than for their own handwritten queries, with an average rating of 4.83 (where 4 is more than half the query, and 5 a complete match within the candidate region).

Table 4.3 Testing: Handwritten Queries for 10 participants (20 queries each)

Top	User Evaluation (1-5)		Match Metrics (%)			
	μ	σ	$\mu(P_{match})$	$\sigma(P_{match})$	$\mu(A_{match})$	$\sigma(A_{match})$
1	2.06	0.63	38.6	11.7	26.7	13.3
5	2.97	0.77	54.9	14.2	39.8	13.8
10	3.15	0.71	63.2	14.9	43.3	14.0

Table 4.4 Testing: 20 Original Query Images

Top	User Evaluation (1-5)		Match Metrics (%)
	μ	σ	$\mu(A_{match})$
1	4.65	0.08	90
5	4.83	0.05	90
10	4.83	0.05	90

Figure 4.3 shows two cartograms based on users and queries respectively. We can notice that the handwritten queries from user 7 have the worst results among all 10

users and 12 of 20 queries are ranked with 1. In the set of handwritten writings of user 7, most math expressions are written in a very compact style, which leads to a poor and inaccurate segmentation of X-Y cut. In this case, most desired regions in the images are not selected as candidates because the query have much smaller size than they should have and so they are not included in the search range. The lower part of the figure shows the result from another aspect: the queries. Query 11 has the worsting results among all 20 queries. we observe that query 11 is a text-like math expression, which means all the symbols in the expression stand in a line. For this kind of math expressions, the X-Y is simple and have similar structure with regular text content, increasing the difficulty of identification [49].

4.5 Discussion

The experimental results reveal how the quality of queries can affect the precision of math retrieval. As shown in Table 4.4, the original query can reach a %90 $\mu(A_{match})$ precision, which means only two original queries from the twenty are missed by the our method. These two queries are "testing 13" and "testing 18" (Figure A.3). We notice that these two queries have very similar layout structure. And by analyzing the X-Y cut applied on these two expressions, we found that these two math expressions actually split into neighboring content after X-Y cut (Figure 4.4), which leads to a poor similarity ranking. However, all other original queries are returned as rank one, indicating that for almost all original queries, the DTW matching algorithm provides very accurate evaluation as long as the X-Y cut can cut the whole expression as one node. For the handwritten queries, the retrieval precision highly depends on the quality of writing. On one hand, as shown in Figure 4.3 (a), some queries such as user 4's queries have very good retrieval rate with a average ranking 3.95, while some

queries such as user 7's queries work very bad with a average ranking 1.85. The reason for this diversity is that the quality of writing can determine the quality of X-Y cut directly. For some handwritten queries, the X-Y cut doesn't work at all, which means these X-Y trees of such queries have very limited similarity with those of their corresponding regions in documents. On the other hand, Figure 4.3 (b) shows the query 11 has the worst retrieval rate among the 20 queries. We can see this query in Figure A.4 (k), the math expression has similar structure with text content in the page, so the X-Y tree for this query is not distinctive among other X-Y trees in the page (Figure 4.5 (a)). On the contrary, Figure A.4 (e) (query 05) has the best retrieval rate (Figure 4.5 (b)) because its unique layout structure. In general, there two dominant factors for missing targets: the quality of X-Y cut and the uniqueness of the tree structure.

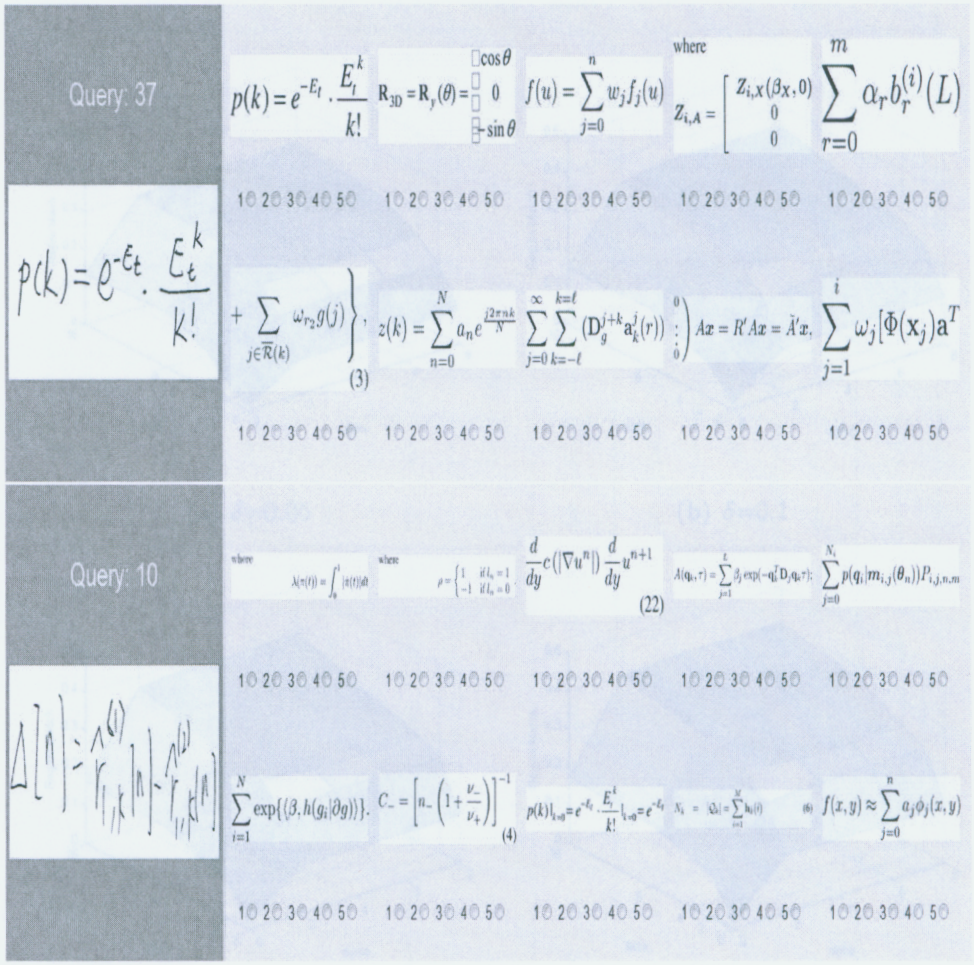
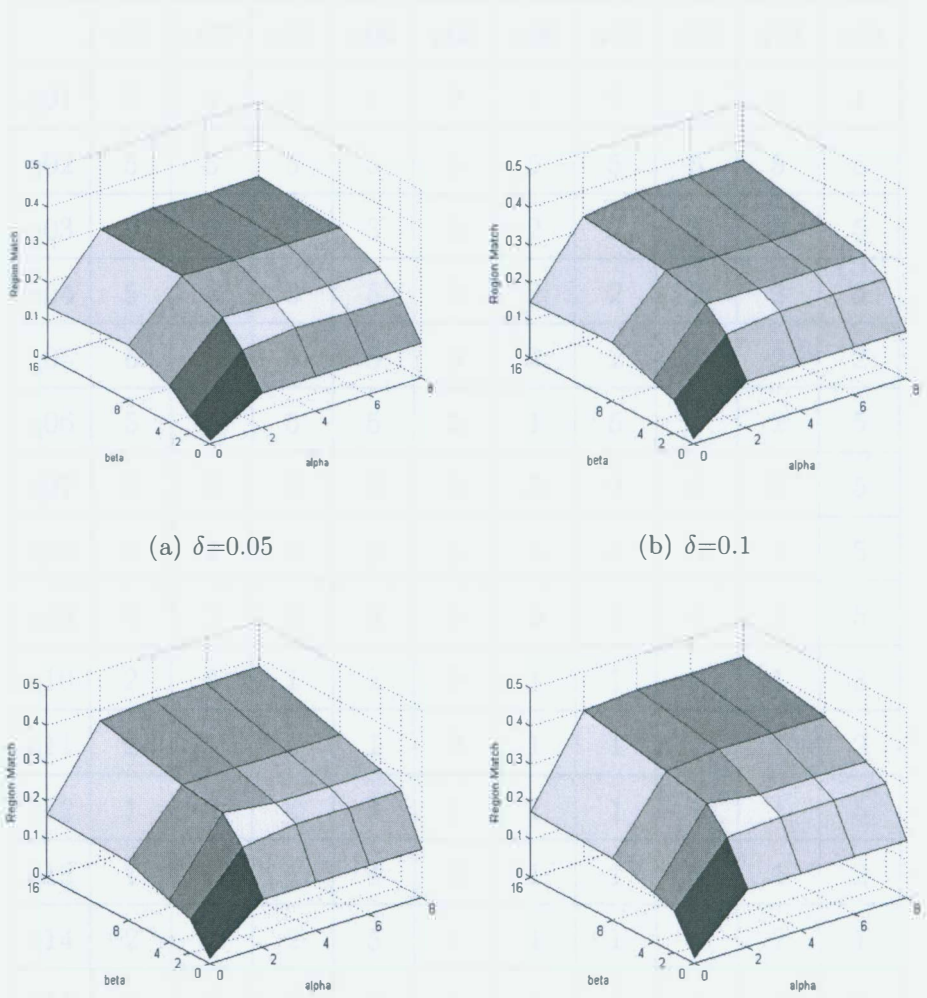


Figure 4.1 Example interface showing search results for handwritten queries. The query is shown at left, and candidates are listed in decreasing rank top-down, left-to-right (the strongest match is shown at top-left). At top is a good result, and a weak one at bottom. Participants could click on images to see them in a separate frame.



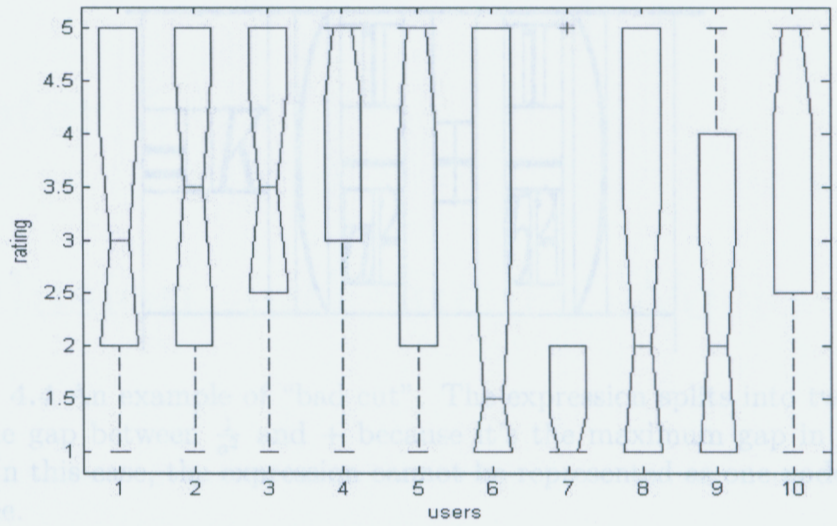
(a) δ=0.05 (b) δ=0.1

(c) δ=0.15 (d) δ=0.2

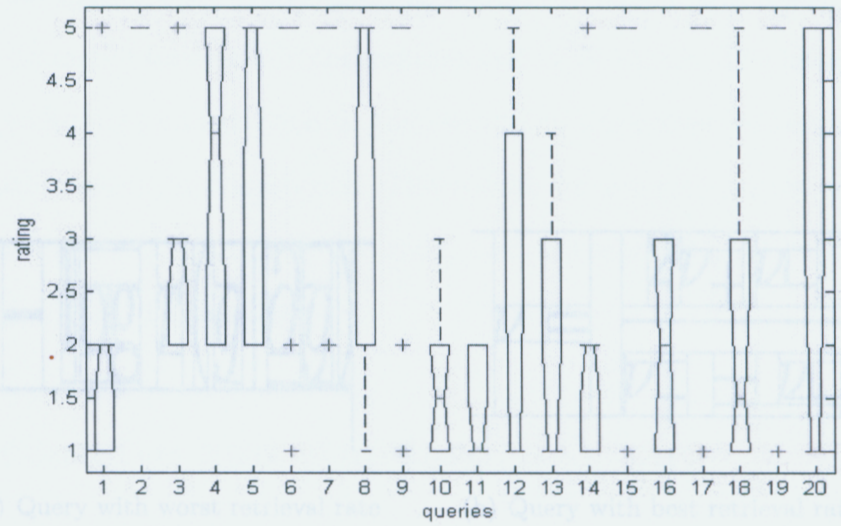
Figure 4.2 Changes in average maximum original query image region match (A_{match}) for given values of α, β and δ.

Table 4.5 Detailed results from user evaluation $\alpha = 4, \beta = 16, \delta = 0.2$ top10 handwritten

	u01	u02	u03	u04	u05	u06	u07	u08	u09	u10
q01	2	5	2	1	2	1	2	1	2	1
q02	5	5	5	5	5	5	5	5	5	5
q03	3	3	3	3	2	2	2	3	3	5
q04	5	5	3	5	5	2	2	2	3	5
q05	5	5	5	5	5	2	2	5	2	2
q06	5	5	5	5	5	1	5	5	2	5
q07	5	5	5	5	5	5	2	5	5	5
q08	1	2	5	5	5	5	5	5	1	5
q09	5	2	5	5	5	5	1	5	5	5
q10	2	3	1	2	2	1	1	1	1	3
q11	2	1	1	1	2	1	1	1	1	2
q12	1	4	4	4	1	1	1	1	1	5
q13	1	3	1	4	4	1	1	1	1	2
q14	2	2	2	5	1	1	1	2	2	1
q15	5	5	5	5	5	5	1	1	5	5
q16	3	2	3	1	2	1	1	1	2	3
q17	5	5	5	5	5	5	1	5	5	5
q18	2	2	3	3	1	1	1	1	1	5
q19	5	5	5	5	5	5	1	5	1	5
q20	1	1	3	5	5	1	1	1	1	5



(a) By users



(b) By queries

Figure 4.3 Distribution of ratings for handwriting queries in testing

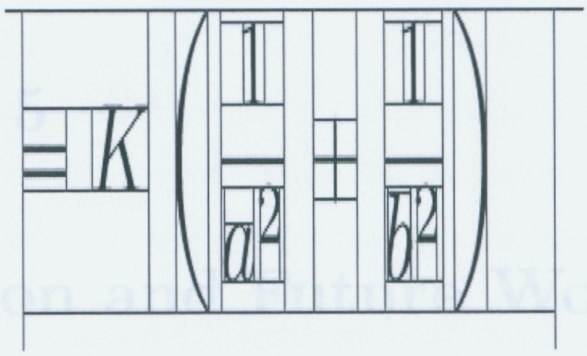
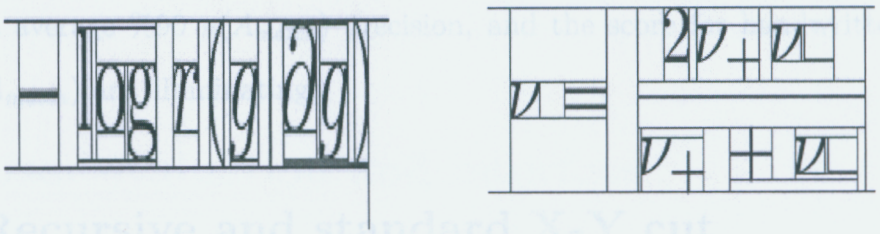


Figure 4.4 An example of "bad cut". The expression splits into two parts from the gap between $\frac{1}{a^2}$ and $+$ because it's the maximum gap in current block. In this case, the expression cannot be represented as one node in the X-Y tree.



(a) Query with worst retrieval rate (b) Query with best retrieval rate

Figure 4.5 The expression structure affects the retrieval rate significantly. The expression on the left has a "text-like" structure while the one on the right has an unique structure

Chapter 5

Conclusion and Future Work

One can use handwritten queries to retrieve math expressions from document databases using page segmentation and image similarity algorithms, by which optical character recognition can be avoided. The results from our experiments proved the feasibility of using visual features for math spotting within document databases. Structural features are extracted successfully and represented by X-Y tree. The image similarity comparison technology called dynamic time wrapping has proven to be useful to compare the math content in documents. The results for original queries are exciting, reaching a average %90 $\mu(A_{match})$ precision, and the score for handwritten queries %43.3 $\mu(A_{match})$ are illuminating.

5.1 Recursive and standard X-Y cut

Recursive X-Y is proven to be more effective at locating expressions than standard X-Y cut when used in our system. Standard X-Y cut is less effective at accommodating structures such as regions than span multiple text columns, making difficult to set thresholds to cut the page. When we tested the two cut algorithms on UW3 database,

the desired math expression are usually mixed with other content under standard X-Y cut.

However, the standard X-Y tree has less depth than the recursive X-Y tree on same images. The shape of recursive X-Y tree is determined by the gaps among components and provides less structural information. So it's not wise for us to compare images by using tree structures provided by recursive X-Y. In our approach, we use the size and depth of nodes in trees for comparison. Standard X-Y cut is used in smaller regions to move out nodes that have obviously different layout structure to the query at the top-level of cutting.

5.2 Tolerance parameters

The parameters used to control the range of search are hard to set, as they depend highly on the quality of queries. For example, original queries almost don't need any tolerance since they are exactly the same as the target. We also notice that the size of the nodes need more tolerance than the depth of nodes. It's reasonable because the size of nodes varies with respect to the depth from linear to quadratic in a binary tree.

5.3 User evaluation and interface

Users are usually invited to evaluate the performance of an information retrieval system. For image retrieval system such like ours, the interface is very important and affects people's judgement to some extent. As shown in the experiment results, the subjectivity of human usually let them have higher grading than machine. Human raters also could assess similarity of matches for similar regions not on the page

containing the image from which a query was defined, which may also account for the ratings produced. In addition, the interface may provide a better way for people to express queries and interact with the system. For example, an interface with recognition function converts users' handwritten queries into typeset or gives them options based on their writing, which works as a normalization system and so improves the retrieval performance.

5.4 Handwritten Queries

Our approach has been proven very effective on original queries (90 percent for maximum region match). For this type of queries, most missing pages are caused by bad cutting which leads to a situation that the targets are not stored in the index system. Once the target exist in the index list, the searching and matching methods can find the original query accurately. The precision for handwritten queries indicates two problems of our system. The first one is the limitation of X-Y cut, it can't provide good tree structure when skew occurs or the letters in the writing overlap. We notice that the handwritten queries with worst retrieval precision are usually written in a compact style. The second problem is that the tolerance value need to increase to adapt irregular queries, which leads to a increasing number of image matching. In the test, each query needs to take an average 45 * 170 = 7650 matchings for one search. The time is not acceptable for searching in real digital database which typically contains more than ten thousands of documents.

5.5 Future Work

- We can consider to improve the cutting accuracy for math expressions since the retrieval precision depends highly on the quality of X-Y cut. Several other X-Y cut strategies might be implemented to have a better segmentation.
- The relationship among the nodes in the X-Y tree can be investigates to reduce the number of candidates and so improve the speed of the program. So statistical models such as Bayesian Networks might be very helpful for this purpose.
- Other features such as global feature and context feature might be explored to improve the efficiency of current method, considering that DTW is computationally expensive despite its high accuracy.
- An conversion between the handwritten queries and the queries used for spotting might be build to work as a preprocessing step to improve the retrieval precision, since the original queries have much better performance than handwritten ones.

Bibliography

[1] R. Datta, D. Joshi, J. Li, and J. Z. Wang, "Image Retrieval: Ideas, Influences, and Trends of the New Age," ACM Computing Surveys **40,** 2 (2008).

[2] N. Vasconcelos, "From Pixels to Semantic Spaces: Advances in Content-Based Image Retrieval," Computer **20,** 20–26 (2007).

[3] J. Ha, R. M. Haralick, and I. T. Phillips, "Recursive x-y cut using bounding boxes of connected components," Proceedings of the Third International Conference on Document Analysis and Recognition **2,** 952 (1995).

[4] G. Nagy and S. Seth, "Hierarchical representation of optically scanned documents," Proc. of ICPR pp. 347–349 (1984).

[5] T. Rath and R. Manmatha, "Word image matching using dynamic time warping," IEEE Computer Society Conference on Computer Vision and Pattern Recognition **2,** 18–20 (2003).

[6] G. Salton, "Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer," Addison-Wesley, Reading (1989).

[7] T. Kanahori and M. Suzuki, "Refinement of digitized documents through recognition of mathematical formulae," In Proceedings of the 2nd International Workshop on Document Image Analysis for Libraries pp. 27–28 (2006).

[8] M. J. Swain and D. H. Ballard, "Color indexing," International Journal of Computer Vision **7,** 632–639 (1991).

[9] G. Joutel, V. Eglin, S. Bres, and H. Emptoz, "Curvelets Based Queries for CBIR Application in Handwriting Collections," Ninth International Conference on Document Analysis and Recognition **2,** 649–653 (2007).

[10] T. Zhao, J. Lu, Y. Zhang, and Q. Xiao, "Feature Selection Based on Genetic Algorithm for CBIR," Congress on Image and Signal Processing **2,** 495–499 (2008).

[11] S. Rudinac, G. Zajic, M. Rudinac, and B. Reljin, "Comparison of CBIR Systems with Different Number of Feature Vector Components," Second International Workshop on Semantic Media Adaptation and Personalization **1,** 199–204 (2007).

[12] Z. Lei, L. Fuzong, and Z. Bo, "A CBIR method based on color-spatial feature," Proceedings of the IEEE, Region 10 Conference **1,** 166–169 (1999).

[13] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger., "The R*-tree: An efficient and robust access method for points and rectangles.," Proceedings of the ACM SIGMOD Conference **2,** 322–331 (1990).

[14] M. Safar and S. Shahabi, C, "Image retrieval by shape: a comparative study," IEEE International Conference on Multimedia and Expo, **1,** 141–144 (2000).

[15] T. Wang, W. Liu, J. Sun, and H. Zhang, "Using fourier descriptor to recognize object's shape," Journal of Computer Research and Development **39,** 411–424 (2002).

[16] A. Folkers and H. Samet, "Content-based image retrieval using Fourier descriptors on a logo database," 16th International Conference on Pattern Recognition **3,** 521–524 (2002).

[17] K. Jalaja, C. Bhagvati, B. L. Deekshatulu, and A. K. Pujari, "Texture element feature characterizations for CBIR," IEEE Proceedings on Geoscience and Remote Sensing Symposium **2,** 4–11 (2005).

[18] M. Do and M. Vetterli, "Wavelet-based texture retrieval using generalized Gaussian densityand Kullback-Leibler distance," IEEE Transactions on Image Processing **11,** 146–158 (2002).

[19] L.-J. Hove, "Evaluating Use of Interfaces for Visual Query Specification," IEEE Transactions on Pattern Analysis and Machine Intelligence **22,** 1349–1380 (2000).

[20] R. Dong, S. Du, and M. Huang, "A semantic retrieval approach by color and spatial location of image regions," Congress on Image and Signal Processing **2,** 466–470 (2008).

[21] N. Rasiwasia and N. Vasconcelos, "A study of query by semantic example," IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops **1,** 1–8 (2008).

[22] S. Rudinac, G. Zajic, M. Rudinac, and B. Reljin, "Content-Based Image Retrieval at the End of the Early Years," IEEE Transactions on Pattern Analysis and Machine Intelligence **22,** 1349–1380 (2000).

[23] J. Li and J. Wang, "Real-time Computerized Annotation of Pictures," IEEE Transactions on Pattern Analysis and Machine Intelligence **30,** 6 (2008).

[24] J. Li and J. Z. Wang, "Automatic Linguistic Indexing of Pictures by a Statistical Modeling Approach," IEEE Transactions on Pattern Analysis and Machine Intelligence **25,** 1075–1088 (2003).

[25] D. Joshi, J. Li, and J. Wang, "Image Classification by a Two Dimensional Hidden Markov Model," IEEE TRANSACTIONS ON IMAGE PROCESSING **3,** 158–172 (2006).

[26] R. Patino-Escarcina and J. Ferreira Costa, "The Semantic Clustering of Images and Its Relation with Low Level Color Features," IEEE International Conference on Semantic Computing **4,** 74 – 79 (2008).

[27] N. Vasconcelos, "Minimum Probability of Error Image Retrieval," IEEE Transactions on Signal Processing **52,** 2322–2336 (2004).

[28] R. Maree, P. Geurts, and L. Wehenkel, "Content-based Image Retrieval by Indexing Random Subwindows with Randomized Trees," Proc. 8th Asian Conference on Computer Vision **4844,** 611–620 (2007).

[29] T. Nakai, K. Kise, and M. Iwamura, "Hashing with Local Combinations of Feature Points and Its Application to Camera-Based Document Image Retrieval," Proc. of the Camera-Based Document Analysis and Recognition **5,** 123–135 (2005).

[30] D. Liu, K. A. Hua, K. Vu, and N. Yu, "Fast Query Point Movement Techniques for Large CBIR Systems," IEEE Transactions on Knowledge and Data Engineering **1,** 243–252 (2008).

[31] H. S. Baird, D. Lopresti, B. D. Davison, and W. M, "Robust document image understanding technologies," In Proceedings of the 1st ACM workshop on Hardcopy document processing **1,** 9–14 (2004).

[32] D. Doermann, "The indexing and retrieval of document images: a survey," Computer Vision and Image Understanding **70,** 17–24 (1998).

[33] A. Kawtrakul and C. Yingsaeree, "A Unified Framework for Automatic Metadata Extraction from Electronic Document," In Proceedings of the International Advanced Digital Library Conference (IADLC) **1,** 231–238 (2005).

[34] D. Doermann and S. Yao, "Generating synthetic data for text analysis systems," In Symposium on Document Analysis and Information Retrieval **1,** 449–467 (1995).

[35] G. D. Silva and J. Hull, "Proper noun detection in document images," Pattern Recognition **27,** 311–320 (2004).

[36] F. Chen, L. Wilcox, and D. Bloomberg, "Detecting and locating partially specified keywords in scanned images using hidden Markov models," In Proceedings of the International Conference on Document Analysis and Recognition **1,** 133–138 (1998).

[37] F. Chen and D. Bloomberg, "Extraction of thematically relevant text from images," In Symposium on Document Analysis and Information Retrieval **1,** 163–178 (2001).

[38] P. Herrmann and G. Schlagetar, "Retrieval of document images using layout knowledge," In Proceedings of the International Conference on Document Analysis and Recognition **1,** 537–540 (2003).

[39] H. Djean and J.-L. Meunier, "Structuring documents according to their table of contents," In Proceedings of the 2005 ACM symposium on Document engineering **1,** 2–4 (2005).

[40] Shafait, D. Keysers, and T. M. Breuel, "Performance Comparison of Six Algorithms for Page Segmentation," Workshop in Document Analysis System **1,** 18–27 (2006).

[41] J. Cullen, J. Hull, and P. Hart, "Document image database retrieval and browsing using texture analysis," In Proceedings of the International Conference on Document Analysis and Recognition **1,** 718–721 (1997).

[42] J. Hull, "Document image matching and retrieval with multiple distortion - invariant descriptors," In International Workshop on Document Analysis Systems **1,** 383–400 (2004).

[43] R. Srihari, "Automatic indexing and content-based retrieval of captioned photographs," In Proceedings of the International Conference on Document Analysis and Recognition **1,** 1165–1168 (1998).

[44] Y. Zheng, S. Member, H. Li, and D. Doermann, "Machine Printed Text and Handwriting Identification in Noisy Document Images," IEEE Trans. Pattern Analysis Machine Intelligence 26 (2004).

[45] F. Shafait, D. Keysers, and T. Breuel, "Performance Evaluation and Benchmarking of Six-Page Segmentation Algorithms," IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (2008).

[46] S. Kane, A. Lehman, and E. Partridge, "Indexing GeorgeWashingtons Handwritten Manuscripts," Technical Report MM-34, Center for Intelligent Information Retrieval (2001).

[47] M. Kolountzakis and K. Kutulakos, "Fast computationion of the Euclidean distance map for binary images," Information Processing Letters **43,** 181–184 (1992).

[48] I. Phillips, "Methodologies for Using UW Databases for OCR and Image Understanding Systems," Proc. Document Recognition V 3305 (1998).

[49] A. Kacem, A. Belaid, and M. Ben Ahmed, "Automatic extraction of printed mathematical formulas using fuzzy logic and propagation of context," International Journal on Document Analysis and Recognition **4,** 97–108 (2001), 10.1007/s100320100064.

Appendix A

Queries for experiments

$$p(d) = \frac{1}{Z} \prod_{ww' \in E} \phi_{ww'}(d_w, d_{w'}),$$

(a) training 01

$$\mathcal{M} = \bigcup_{t \in [a,b]} C_t,$$

(b) training 02

$$p_2(k) = \frac{P}{K} p_1(k)$$

(c) training 03

$$s = \bar{s} + \sum s_i \alpha_i$$

(d) training 04

$$\sum_{x=o}^{L-1} f(x) \cos$$

(e) training 05

$$G = \begin{pmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{pmatrix}$$

(f) training 06

$$q(k,i,j) = \ln\left(\frac{h_i^k + \epsilon}{h_j^k + \epsilon}\right)$$

(g) training 07

$$Z = F \frac{B_{,} - S_{,}}{d_{,}}$$

(h) training 08

$$M_V(i,j) = \sum_{m=2}^{|V|-1} D_{V,m}(i,j)$$

(i) training 09

$$f_n^*(\mathcal{I}, u) = \left| u - \hat{x}_n^* \right|$$

(j) training 10

$$\frac{1}{\sum_{n=0}^{N-1} |\boldsymbol{S}_n|}$$

(k) training 11

$$\frac{\partial X}{\partial t} = X^{k+1} - X^k.$$

(l) training 12

$$\frac{1}{F} \sum_{i=1}^{F} \frac{k_{ci}}{k}$$

(m) training 13

$$\frac{m_p + 1}{m_p + n_p + 3}$$

(n) training 14

$$V_j(I) = \sum_{f \in F} \Delta V_j(f)$$

(o) training 15

$$\left(1 - \alpha\right) B_t\left(x, y\right)$$

(p) training 16

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(q) training 17

$$\prod_{k=1}^{S} [P(V_k | pa(V_k)) P(M_{V_k} | V_k)]$$

(r) training 18

$$\int_{\mathbf{x}} f(\mathbf{x}) log \frac{f(\mathbf{x})}{g(\mathbf{x})}$$

(s) training 19

$$b(o|S_k) = \mathcal{N}(o|\mu_k, \Sigma_k)$$

(t) training 20

Figure A.1 The original queries for training

(a) training 01 (b) training 02 (c) training 03 (d) training 04

(e) training 05 (f) training 06 (g) training 07 (h) training 08

(i) training 09 (j) training 10 (k) training 11 (l) training 12

(m) training 13 (n) training 14 (o) training 15 (p) training 16

(q) training 17 (r) training 18 (s) training 19 (t) training 20

Figure A.2 The handwritten queries for training from one of the users

$$\frac{\kappa}{4\pi \sinh \kappa} \cosh(\kappa \mu^T \mathrm{x})$$

(a) testing 01

$$f(x,y) \approx \sum_{j=0}^{n} a_j \phi_j(x,y)$$

(b) testing 02

$$\frac{35x^4 - 30x^2 + 3}{8}$$

(c) testing 03

$$\sum_{k=1}^{K} \pi(k) \mathbf{p}_k(x)$$

(d) testing 04

$$\nu = \frac{2\nu_+ \nu_-}{\nu_+ + \nu_-}$$

(e) testing 05

$$u_i = \sum_{x \in S_i} p(x) \in [0,1]$$

(f) testing 06

$$\frac{dc_i}{dt} = -\lambda_i^{\alpha} c_i$$

(g) testing 07

$$\mathrm{f_s} = \frac{1}{2\pi} \int_0^{2\pi} g_\phi \mathrm{f} \, d\phi.$$

(h) testing 08

$$\int_{X \times X} e_\varphi(x,x') dx dx'$$

(i) testing 09

$$A\left(\widehat{W}_A - V_A\right) + N\left(W_N - V_N\right)$$

(j) testing 10

$$-\log r(g|\partial g)$$

(k) testing 11

$$\frac{1}{2\mu_0} \exp\left(-\frac{|\delta(u)|}{\mu_0}\right)$$

(l) testing 12

$$\left(\frac{1}{a^2} + \frac{1}{b^2}\right)$$

(m) testing 13

$$\Delta[n] = \hat{r}_{1,K}^{(1)}[n] - \hat{r}_{1,K}^{(2)}[n],$$

(n) testing 14

$$C^{(+)} = \begin{pmatrix} P & Q \\ Q^T & R \end{pmatrix}^{-1}$$

(o) testing 15

$$\frac{1}{1 + \exp\{Af(x) + B\}}$$

(p) testing 16

$$p(k) = e^{-E_l} \cdot \frac{E_l^k}{k!}$$

(q) testing 17

$$\frac{1}{2}\left(f_i - f_{\hat{j}}\right)$$

(r) testing 18

$$\frac{1}{M} \cdot \mathrm{B} \cdot \underline{\mathrm{P}}(Y_m = n)$$

(s) testing 19

$$g = \begin{pmatrix} 1 & 0 \\ 0 & \sin^2 \theta \end{pmatrix}$$

(t) testing 20

Figure A.3 The original queries for testing

(a) testing 01

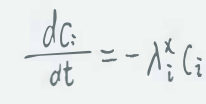
(b) testing 02

(c) testing 03

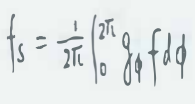
(d) testing 04

(e) testing 05

(f) testing 06



(g) testing 07



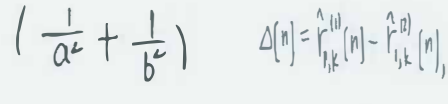
(h) testing 08

(i) testing 09

(j) testing 10

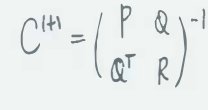
(k) testing 11

(l) testing 12

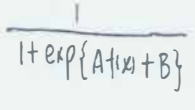


(m) testing 13

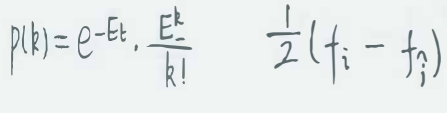
(n) testing 14



(o) testing 15

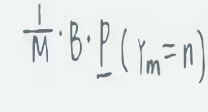


(p) testing 16

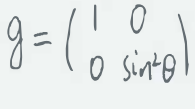


(q) testing 17

(r) testing 18

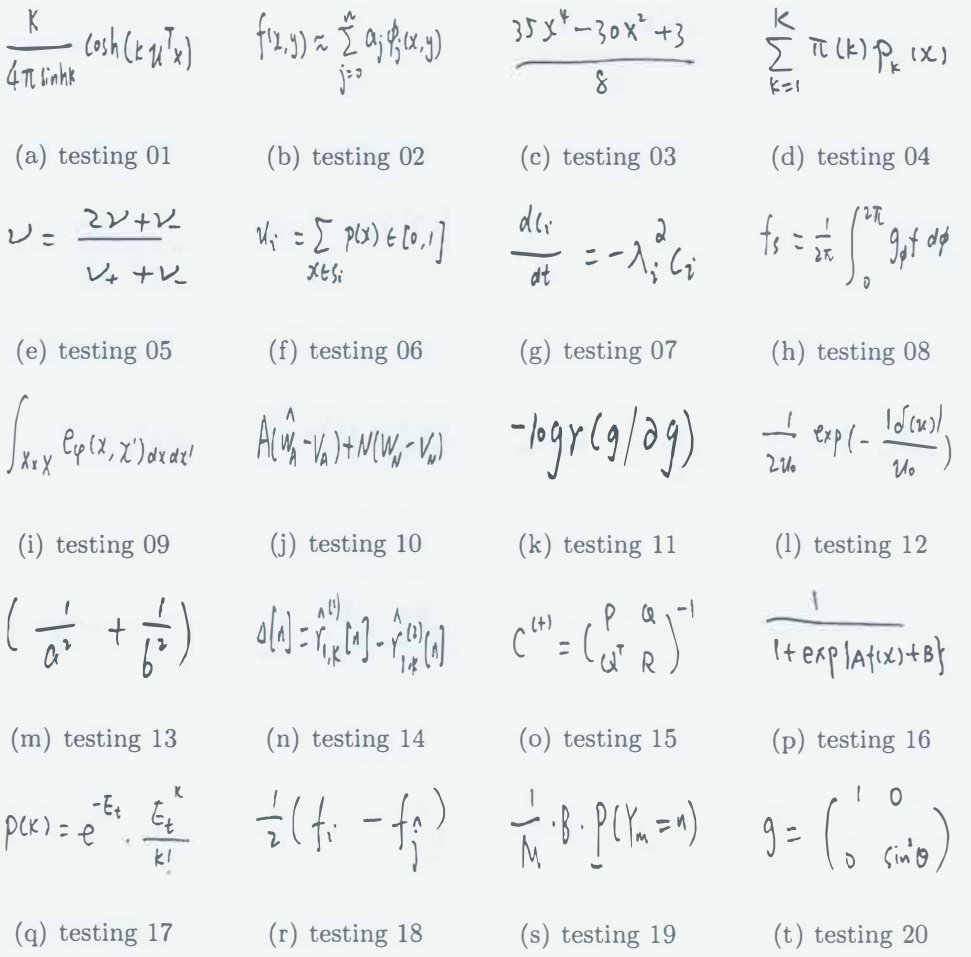


(s) testing 19



(t) testing 20

Figure A.4 The handwritten queries for testing from user 1



(a) testing 01 (b) testing 02 (c) testing 03 (d) testing 04

(e) testing 05 (f) testing 06 (g) testing 07 (h) testing 08

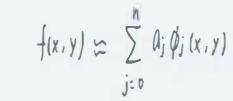
(i) testing 09 (j) testing 10 (k) testing 11 (l) testing 12

(m) testing 13 (n) testing 14 (o) testing 15 (p) testing 16

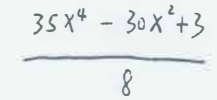
(q) testing 17 (r) testing 18 (s) testing 19 (t) testing 20

Figure A.5 The handwritten queries for testing from user 2

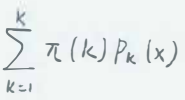
(a) testing 01



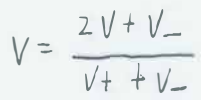
(b) testing 02



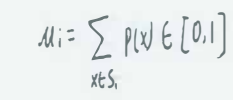
(c) testing 03



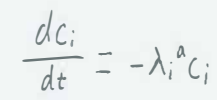
(d) testing 04



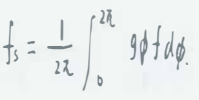
(e) testing 05



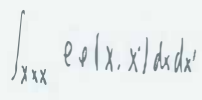
(f) testing 06



(g) testing 07



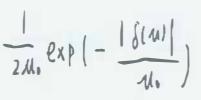
(h) testing 08



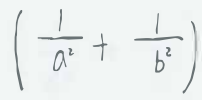
(i) testing 09

(j) testing 10

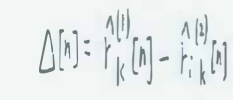
(k) testing 11



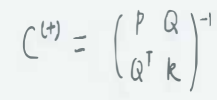
(l) testing 12



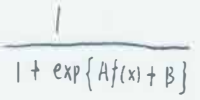
(m) testing 13



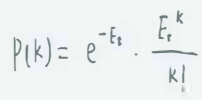
(n) testing 14



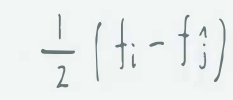
(o) testing 15



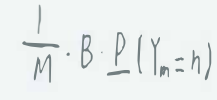
(p) testing 16



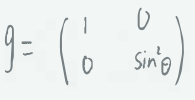
(q) testing 17



(r) testing 18

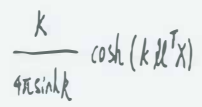


(s) testing 19

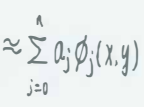


(t) testing 20

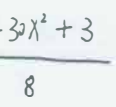
Figure A.6 The handwritten queries for testing from user 3



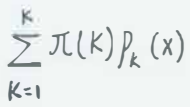
(a) testing 01



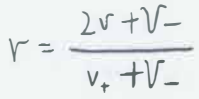
(b) testing 02



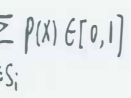
(c) testing 03



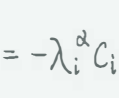
(d) testing 04



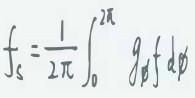
(e) testing 05



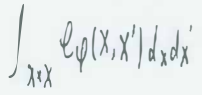
(f) testing 06



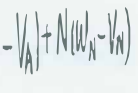
(g) testing 07



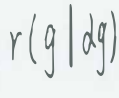
(h) testing 08



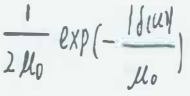
(i) testing 09



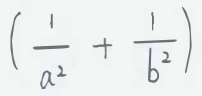
(j) testing 10



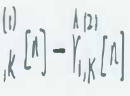
(k) testing 11



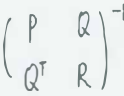
(l) testing 12



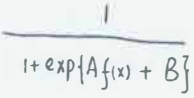
(m) testing 13



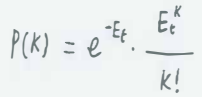
(n) testing 14



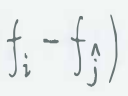
(o) testing 15



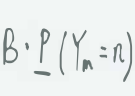
(p) testing 16



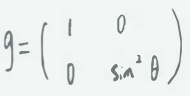
(q) testing 17



(r) testing 18

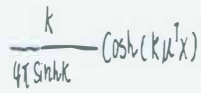


(s) testing 19

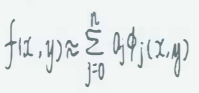


(t) testing 20

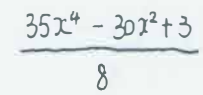
Figure A.7 The handwritten queries for testing from user 4



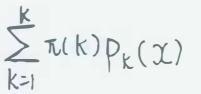
(a) testing 01



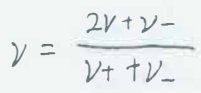
(b) testing 02



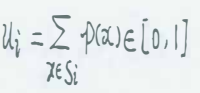
(c) testing 03



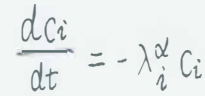
(d) testing 04



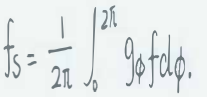
(e) testing 05



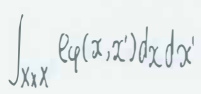
(f) testing 06



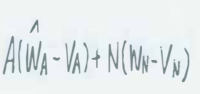
(g) testing 07



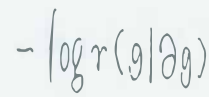
(h) testing 08



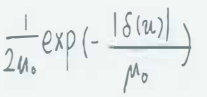
(i) testing 09



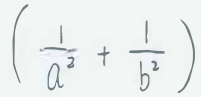
(j) testing 10



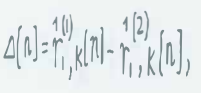
(k) testing 11



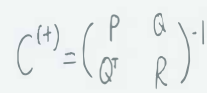
(l) testing 12



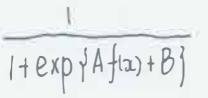
(m) testing 13



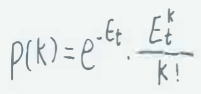
(n) testing 14



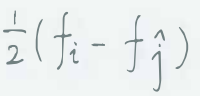
(o) testing 15



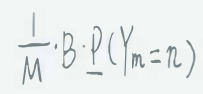
(p) testing 16



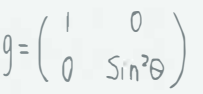
(q) testing 17



(r) testing 18

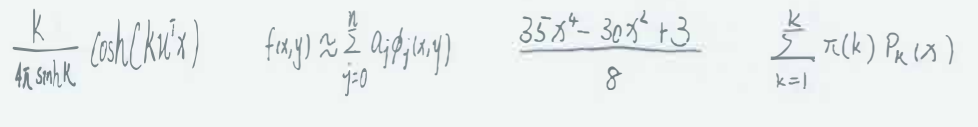


(s) testing 19



(t) testing 20

Figure A.8 The handwritten queries for testing from user 5

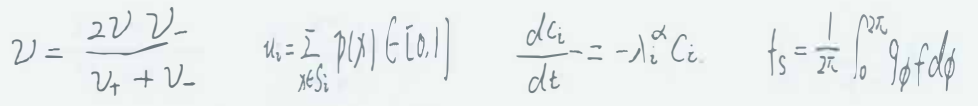


(a) testing 01

(b) testing 02

(c) testing 03

(d) testing 04



(e) testing 05

(f) testing 06

(g) testing 07

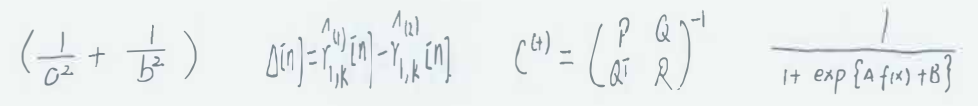
(h) testing 08

(i) testing 09

(j) testing 10

(k) testing 11

(l) testing 12

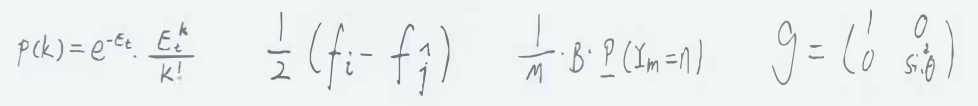


(m) testing 13

(n) testing 14

(o) testing 15

(p) testing 16



(q) testing 17

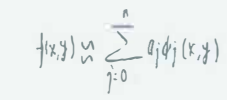
(r) testing 18

(s) testing 19

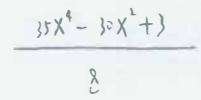
(t) testing 20

Figure A.9 The handwritten queries for testing from user 6

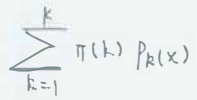
(a) testing 01



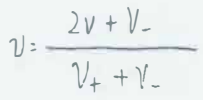
(b) testing 02



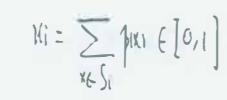
(c) testing 03



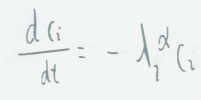
(d) testing 04



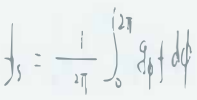
(e) testing 05



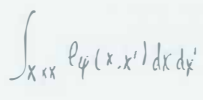
(f) testing 06



(g) testing 07



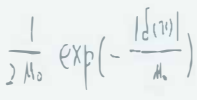
(h) testing 08



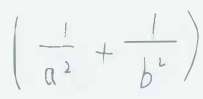
(i) testing 09

(j) testing 10

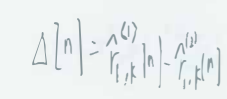
(k) testing 11



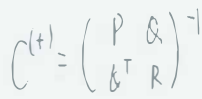
(l) testing 12



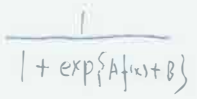
(m) testing 13



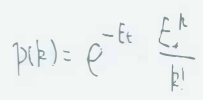
(n) testing 14



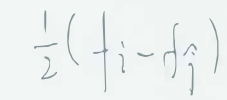
(o) testing 15



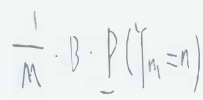
(p) testing 16



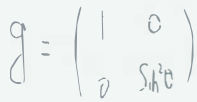
(q) testing 17



(r) testing 18

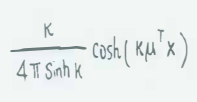


(s) testing 19



(t) testing 20

Figure A.10 The handwritten queries for testing from user 7

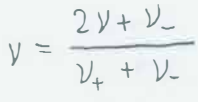


(a) testing 01

(b) testing 02

(c) testing 03

(d) testing 04

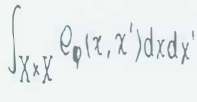


(e) testing 05

(f) testing 06

(g) testing 07

(h) testing 08

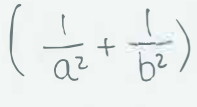


(i) testing 09

(j) testing 10

(k) testing 11

(l) testing 12

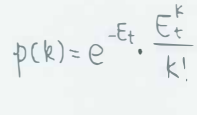


(m) testing 13

(n) testing 14

(o) testing 15

(p) testing 16



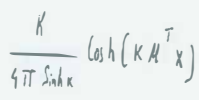
(q) testing 17

(r) testing 18

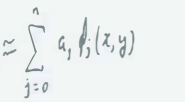
(s) testing 19

(t) testing 20

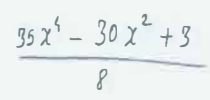
Figure A.11 The handwritten queries for testing from user 8



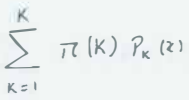
(a) testing 01



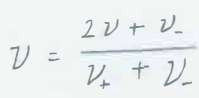
(b) testing 02



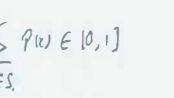
(c) testing 03



(d) testing 04

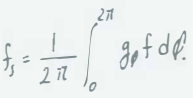


(e) testing 05

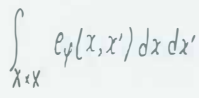


(f) testing 06

(g) testing 07



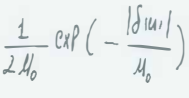
(h) testing 08



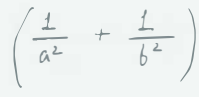
(i) testing 09

(j) testing 10

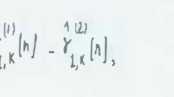
(k) testing 11



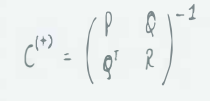
(l) testing 12



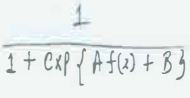
(m) testing 13



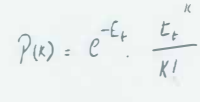
(n) testing 14



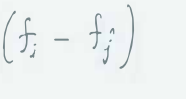
(o) testing 15



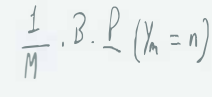
(p) testing 16



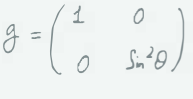
(q) testing 17



(r) testing 18



(s) testing 19



(t) testing 20

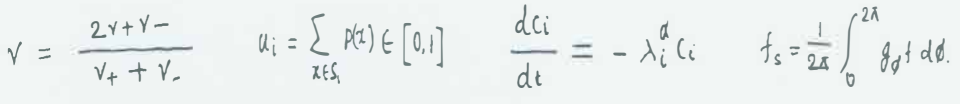
Figure A.12 The handwritten queries for testing from user 9

(a) testing 01

(b) testing 02

(c) testing 03

(d) testing 04

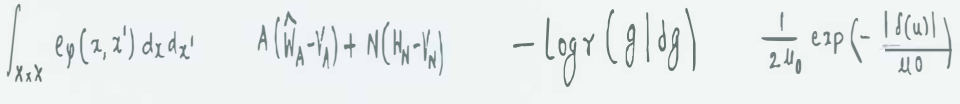


(e) testing 05

(f) testing 06

(g) testing 07

(h) testing 08



(i) testing 09

(j) testing 10

(k) testing 11

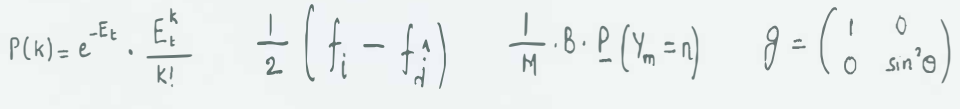
(l) testing 12

(m) testing 13

(n) testing 14

(o) testing 15

(p) testing 16



(q) testing 17

(r) testing 18

(s) testing 19

(t) testing 20

Figure A.13 The handwritten queries for testing from user 10