1997

Multimedia Indexing And Retrieval Research at the Center for Intelligent Information Retrieval

R. Manmatha
University of Massachusetts - Amherst

Multimedia Indexing And Retrieval Research at the Center for Intelligent Information Retrieval

R. Manmatha *
Multimedia Indexing and Retrieval Group
Center for Intelligent Information Retrieval
Computer Science Department
University of Massachusetts, Amherst, MA 01003
manmatha@cs.umass.edu

Abstract

The digital libraries of the future will include not only (ASCII) text information but scanned paper documents as well as still photographs and videos. There is, therefore, a need to index and retrieve information from such multi-media collections. The Center for Intelligent Information Retrieval (CIIR) has a number of projects to index and retrieve multi-media information. These include:

1. *The extraction of text from images which may be used both for finding text zones against general backgrounds as well as for indexing and retrieving image information.*
2. *Indexing hand-written and poorly printed documents using image matching techniques (word spotting).*
3. *Indexing images using their content.*

1 Introduction

The digital libraries of the future will include not only (ASCII) text information but scanned paper documents as well as still photographs and videos. There is, therefore, a need to index and retrieve information from such multi-media collections. The Center for Intelligent Information Retrieval (CIIR) has a number of projects to index and retrieve multi-media information. These include:

1. Finding Text in Images: The conversion of scanned documents into ASCII so that they can be indexed using INQUERY (CIIR's text retrieval engine). Current Optical Character Recognition Technology (OCR) can convert scanned text to ASCII but is limited to good clean machine printed fonts against clean backgrounds. Handwritten text, text printed against shaded or textured backgrounds and text embedded in images cannot be recognized well (if it can be recognized at all) with existing OCR technology. Many financial documents, for example, print text against shaded backgrounds to prevent copying.

   The Center has developed techniques to detect text in images. The detected text is then cleaned up and binarized and run through a commercial OCR. Such techniques can be applied to zoning text found against general backgrounds as well as for indexing and retrieving images using the associated text.

2. Word Spotting: The indexing of hand-written and poorly printed documents using image matching techniques. Libraries hold vast collections of original handwritten manuscripts, many of which have never been published. Word Spotting can be used to create indices for such handwritten manuscript archives.

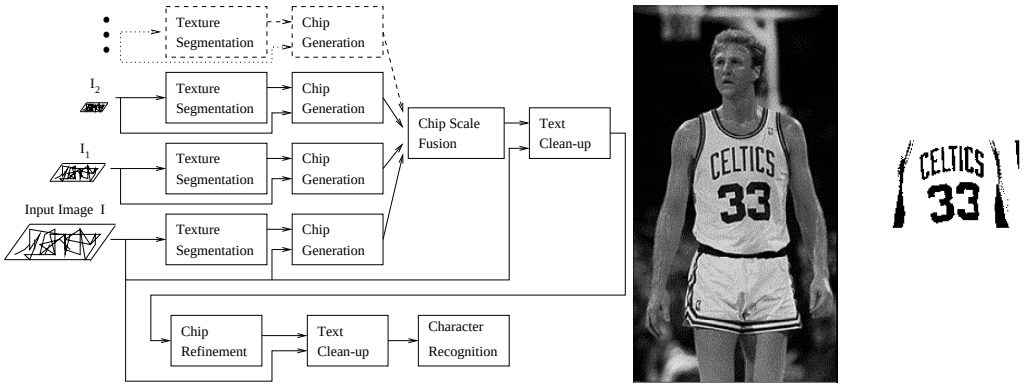
3. Image Retrieval: Indexing images using their content. The Center has also developed techniques to index and retrieve images by color and appearance.

2 Finding Text in Images

Most of the information available today is either on paper or in the form of still photographs and videos. To build digital libraries, this large volume of information needs to be digitized into images and the text converted to ASCII for storage, retrieval, and easy manipulation. For example, video sequences of events such as a basketball game can be annotated and indexed by extracting a player's number, name and the team name that appear on the player's uniform (Figure 1(b, c)). This maybe combined with methods for image indexing and retrieval based on image content (see section 3).

Current OCR technology [1, 20] is largely restricted to finding text printed against clean backgrounds, since

*This material is based on work supported in part by the National Science Foundation, Library of Congress and Department of Commerce under cooperative agreement number EEC-9209623, in part by the United States Patent and Trademarks Office and the Defense Advanced Research Projects Agency/ITO under ARPA order number D468, issued by ESC/AXS contract number F19628-95-C-0235, in part by NSF IRI-9619117 and in part by NSF Multimedia CDA-9502639. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsors.



(a) (b) (c)

Figure 1: The system, example input image, and extracted text. (a) The top level components of the text detection and extraction system. The pyramid of the input image is shown as I, I_1, I_2 ...; (b) An example input image; (c) Output of the system before being fed to the Character Recognition module.

in these cases it is easy to binarize the input images to extract text (text binarization) before character recognition begins. It cannot handle text printed against shaded or textured backgrounds, nor text embedded in pictures. More sophisticated text reading systems usually employ page segmentation schemes to identify text regions. Then an OCR module is applied only to the text regions to improve its performance. Some of these schemes [32, 33, 21, 23] are top-down approaches, some are bottom-up methods [7, 22], and others are based on texture segmentation techniques in computer vision [8]. However, the top-down and bottom-up approaches usually require the input image to be binary and have a Manhattan layout. Although the approach in [8] can in principle be applied to greyscale images, it was only used on binary document images, and in addition, the text binarization problem was not addressed. In summary, few working systems have been reported that can read text from document pages with both structured and non-structured layouts. A brief overview of a system developed at CIIR for constructing a complete automatic text reading system is presented here (for more details see [34, 35]).

2.1 System Overview

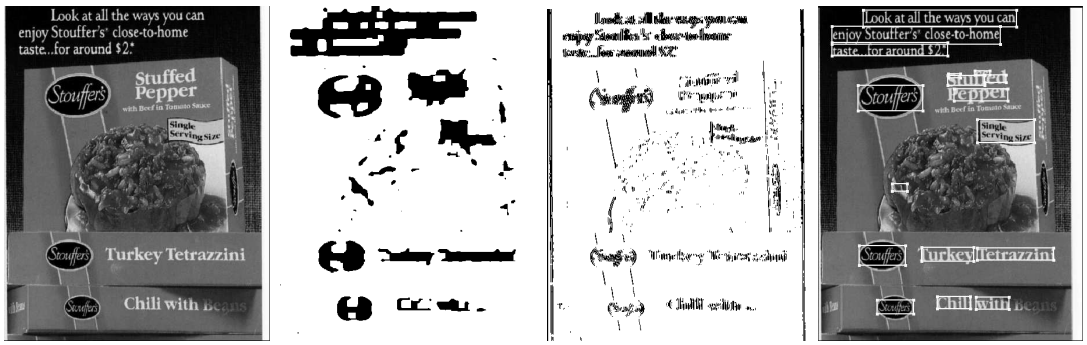
The system takes advantage of the following distinctive characteristics of text which make it stand out from other image information: (1) Text possesses a distinctive frequency and orientation attributes; (2) Text shows spatial cohesion — characters of the same text string are of similar heights, orientation and spacing.

The first characteristic suggests that text may be treated as a distinctive texture, and thus be segmented out using texture segmentation techniques. Thus, the first phase of our system is Texture Segmentation as shown in Figure 1(a). In the Chip Generation phase, strokes are extracted from the segmented text regions. Using reasonable heuristics on text strings based on the second characteristic, the extracted strokes are then processed to form tight rectangular bounding boxes around the corresponding text strings. To detect text over a wide range of font sizes, the above steps are applied to a pyramid of images generated from the input image, and then the boxes formed at each resolution level of the pyramid are fused at the original resolution. A Text Clean-up module which removes the background and binarizes the detected text is applied to extract the text from the regions enclosed by the bounding boxes. Finally, text bounding boxes are refined (re-generated) by using the extracted items as strokes. These new boxes usually bound text strings better. The Text Clean-up process is then carried out on the regions bounded by these new boxes to extract cleaner text, which can then be passed through a commercial OCR engine for recognition if the text is of an OCR-recognizable font. The phases of the system are discussed in the following sections.

2.2 The Texture Segmentation Module

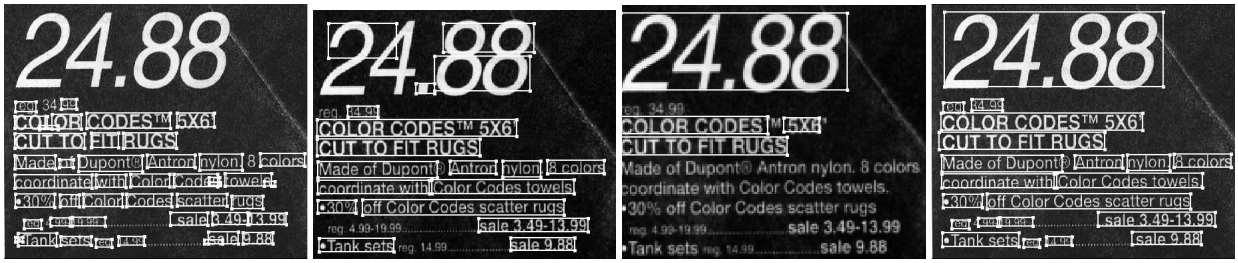
A standard approach to texture segmentation is to first filter the image using a bank of linear filters such as Gaussian derivatives [11] or Gabor functions, followed by some non-linear transformation such as a hyperbolic function $tanh(\alpha t)$. Then features are computed to form a feature vector for each pixel from the filtered images. These feature vectors are then classified to segment the textures into different classes (for more details see [34, 35]).

Figure 2(a) shows a portion of an original input image with a variety of textual information to be extracted. There is text on a clean dark background, text printed on Stouffer boxes, Stouffer's trademarks (in script), and a picture of the food. Figure 2(b) shows the final segmented text regions.



(a) (b) (c) (d)

Figure 2: Results of Texture Segmentation and Chip Generation. (a) Portion of an input image; (b) The final segmented text regions; (c) Extracted strokes; (d) Text chips mapped on the input image.



(a) (b) (c) (d)

Figure 3: The scale problem and its solution. (a) Chips generated for the input image at full resolution; (b) half resolution; (c) $\frac{1}{4}$ resolution; (d) Chips generated at all three levels mapped onto the input image. Scale-redundant chips are removed.

2.3 The Chip Generation Phase

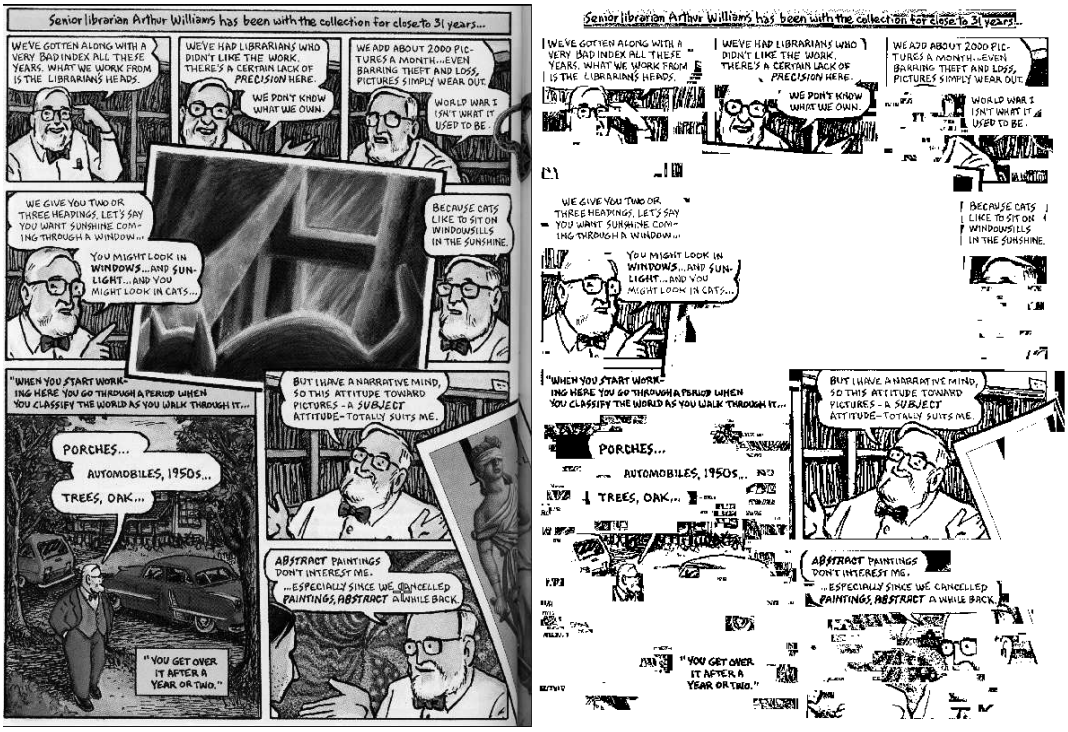
In practice, text may occur in images with complex backgrounds and texture patterns, such as foliage, windows, grass etc. Thus, some non-text patterns may pass the filters and initially be misclassified as text (Figure 2(b)). Furthermore, segmentation accuracy at texture boundaries is a well-known and difficult problem in texture segmentation. Consequently, it is often the case that text regions are connected to other regions which do not correspond to text, or one text string might be connected to another text string of a different size or intensity. This might cause problems for later processing. For example, if two text strings with significantly different intensity levels are joined into one region, one intensity threshold might not separate both text strings from the background.

Therefore, heuristics need to be employed to refine the segmentation result. Since the segmentation process usually finds text regions while excluding most of those that are non-text, these regions can be used to direct further processing (**focus of attention**). Furthermore, since text is intended to be readable, there is usually a significant contrast between it and the background. Thus contrast can be utilized finding text. Also, it is usually the case that characters in the same word/phrase/sentence are of the same font and have similar heights and inter-character spaces. Finally, it is obvious that characters in a horizontal text string are horizontally aligned. Therefore, all the heuristics above are incorporated in the Chip Generation phase in a bottom-up fashion: significant edges form strokes (Figure 2(c)); strokes from the segmented regions are aggregated to form chips corresponding to text strings. The rectangular bounding boxes of the chips are used to indicate where the hypothesized (detected) text strings are (Figure 2(d)). These steps are described in detail in [34, 35].

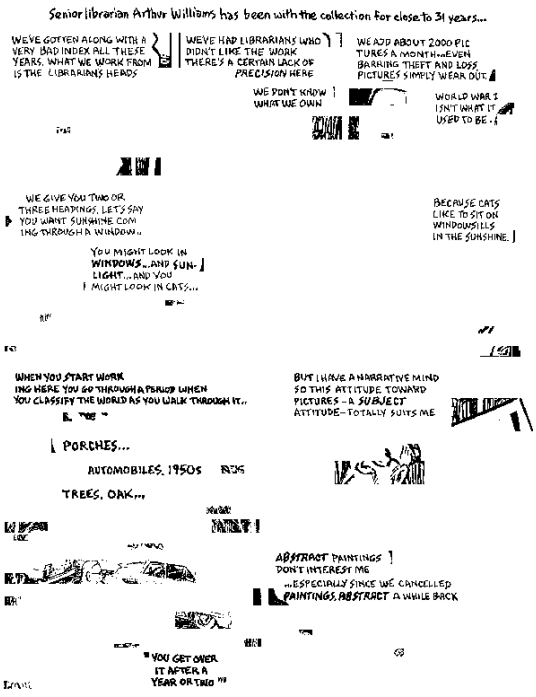
2.4 A Solution to the Scale Problem

The frequency channels used in the segmentation process work well to cover text over a certain range of font sizes. Text from larger font sizes is either missed or fragmented. This is called the **scale problem**. Intuitively, the larger the font size of the text, the lower the frequency it possesses. Thus, when the text font size gets too large, its frequency falls outside the channels selected in section 2.2.

A pyramid approach (Figure 1(a)) is used to solve the scale problem: a pyramid of the input image is formed and each image in the pyramid is processed as described in the previous sections. At the bottom of the pyramid is the original image; the image at each level (other than the bottom) has half of the resolution as that of the im-



(a) (b)



(c)

Figure 4: Binarization results before and after the Chip Refinement step. (a) Input image; (b) binarization result before refinement; (c) after refinement.

age one level below. Text of smaller font sizes can be detected using the images lower in the pyramid (Figure 3(a)), while text of large font sizes is found using images higher in the pyramid (Figure 3(c). The bounding boxes of detected text regions at each level are mapped back to the original input image and the redundant boxes are then removed as shown in Figure 3(d). Details are presented in [34, 35].

2.5 Text on Complex Backgrounds

The previous sections describe a system which detects text in images and puts boxes around detected text strings in the input image. Since text may be printed against complex image backgrounds, which current OCR systems cannot handle well, it is desirable to have the backgrounds removed first. In addition, OCR systems require that the text must be binarized before actual recognition starts. In this system, the background removal and text binarization is done by applying an algorithm to the text boxes individually instead of trying to binarize the input image as a whole. This allows the process to adapt to the individual context of each text string. The details of the algorithm are in [34, 35].

2.6 The Text Refinement

Sometimes non-text items are identified as text as well. In addition, the bounding boxes of the chips sometimes do not tightly surround the text strings. The consequence of these problems is that non-text items may occur in the binarized image, produced by mapping the extracted items onto the original page. An example is shown in Figure 4(a,b). These non-text items are not desirable.

However, by treating the extracted items as strokes, the Chip Refinement module which is essentially similar to the chip Generation module but with stronger constraints, can be applied here to eliminate the non-text items and hence form tighter text bounding boxes. This can be achieved because (1) the clean-up procedure is able to extract most characters without attaching to nearby characters and non-text items (Figure 4(b)), and (2) most of the strokes at this stage are composed of complete or almost complete characters, as opposed to the vertical connected edges of the characters in the initial processing. Thus, it can be expected that the correct text strokes comply more consistently with the heuristics used in the early Chip Generation phase. The significant improvement is clearly shown in 4c.

2.7 Experiments

The system has been tested over 48 images from a wide variety of sources: digitized video frames, photographs, newspapers, advertisements in magazines or sales flyers, and personal checks. Some of the images have regular page layouts, others do not. It should be pointed out that all the system parameters remain the same throughout the entire set of test images, showing the robustness of the system.

Characters and words (as perceived by one of the authors) were counted in each image as ground truth. The total numbers over the whole test set are shown in the "Total Perceived" column in Table 1. The detected characters and words are those which are completely enclosed by the boxes produced after the Chip Scale Fusion step. The total numbers of detected characters and words over the entire test set are shown in the "Total Detected" column. Characters and words clearly readable by a person after the Chip Refinement and Text Clean-up steps (final extracted text) are also counted for each image, with the total numbers shown in the "Total Clean-up" column. The column "Total OCRable" shows the total numbers of cleaned-up characters and words that appear to be of OCR recognizable fonts in 35 of the binarized images. Note that only the text which is horizontally aligned is counted (skew angle of the text string is less than roughly 30 degrees)[1]. The "Total OCRed" column shows the numbers of characters and words from the "Total OCRable" sets correctly recognized by Caere's commercial WordScan OCR engine.

Figure 5(a) is a portion of an original input image which has no structured layout. The final binarization result is shown in (b) and the corresponding OCR output is shown in (c). Notice that most of the text is detected, and most of the text of machine-printed fonts are correctly recognized by the OCR engine. It should be pointed out that the cleaned-up output looks fine to a person in the places where the OCR errors occurred.

3 Word Spotting: Indexing Handwritten Archival Manuscripts

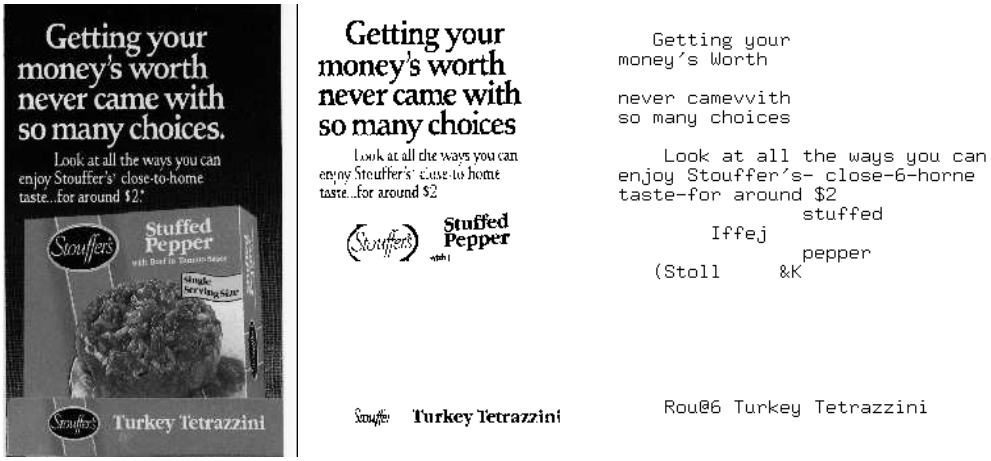
There are many historical manuscripts written in a single hand which it would be useful to index. Examples include the W. B. DuBois collection at the University of Massachusetts, Margaret Sanger's collected works at Smith College and the early Presidential libraries at the Library of Congress. These manuscripts are largely written in a single hand. Such manuscripts are valuable resources for scholars as well as others who wish to consult the original manuscripts and considerable effort has gone into manually producing indices for them. For example, a substantial collection of Margaret Sanger's work has been recently put on microfilm (see http://MEP.cla.sc.edu/Sanger/SangBase.HTM) with an item by item index. These indices were created manually. The indexing scheme described here will help in the automatic creation and production of indices and concordances for such archives.

One solution is to use Optical Character Recognition (OCR) to convert scanned paper documents into ASCII.

[1] Here, the focus is on finding horizontal, linear text strings only. The issue of finding text strings of any orientation will be addressed in future work.

Table 1: Summary of the system's performance. 48 images were used for detection and clean-up. Out of these, 35 binarized images were used for the OCR process.

	Total Perceived	Total Detected	Total Clean-up	Total OCRable	Total OCRed
Char	21820	20788 (95%)	91%	14703	12428 (84%)
Word	4406	4139 (93%)	86%	2981	2314 (77%)



(a) (b) (c)

Figure 5: Example 1. (a) Original image (ads11); (b) Extracted text; (c) The OCR result using Caere's WordScan Plus 4.0 on b.

Existing OCR technology works well with standard machine printed fonts against clean backgrounds. It works poorly if the originals are of poor quality or if the text is handwritten. Since Optical Character Recognition (OCR) does not work well on handwriting, an alternative scheme based on matching the images of the words was proposed by us in [18, 17, 15] for indexing such texts. Here a brief summary of the work is presented.

Since the document is written by a single person, the assumption is that the variation in the word images will be small. The proposed solution will first segment the page into words and then match the actual word images against each other to create equivalence classes. Each equivalence class will consist of multiple instances of the same word. Each word will have a link to the page it came from. The number of words in each equivalence class will be tabulated. Those classes with the largest numbers of words will probably be stopwords, i.e. conjunctions such as "and" or articles such as "the". Classes containing stopwords are eliminated (since they are not very useful for indexing). A list is made of the remaining classes. This list is ordered according to the number of words contained in each of the classes. The user provides ASCII equivalents for a representative word in each of the top m (say m = 2000) classes. The words in these classes can now be indexed. This technique will be called "word spotting" as it is analogous to "word spotting" in speech processing [9].

The proposed solution completely avoids machine recognition of handwritten words as this is a difficult task [20]. Robustness is achieved compared to OCR systems for two reasons:

1. Matching is based on entire words. This is in contrast to conventional OCR systems which essentially recognize characters rather than words.

2. Recognition is avoided. Instead a human is placed in the loop when ASCII equivalents of the words must be provided.

Some of the matching aspects of the problem are discussed here (for a discussion of page segmentation into words, see [18]). The matching phase of the problem is expected to be the most difficult part of the problem. This is because unlike machine fonts, there is some variation in even a single person's handwriting. This variation is difficult to model. Figure (6) shows two examples of the word "Lloyd" written by the same person. The last image is produced by XOR'ing these two images. The white areas in the XOR image indicate where the two versions of "Lloyd" differ. This result is not unusual. In fact, the differences are sometimes even larger.

The performance of two different matching techniques is discussed here. The first, based on Euclidean distance mapping [2], assumes that the deformation between words can be modelled by a translation (shift). The second, based on an algorithm by Scott and Longuet

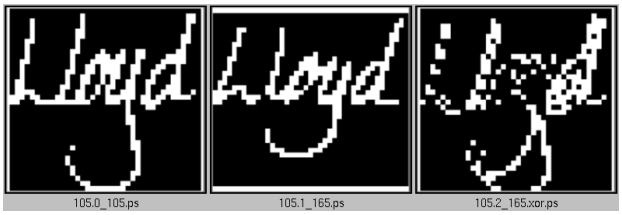


Figure 6: Two examples of the word “Lloyd” and the XOR image

Higgins [28] models the transformation between words using an affine transform.

3.1 Prior Work

The traditional approach to indexing documents involves first converting them to ASCII and then using a text based retrieval engine [30]. Scanned documents printed in standard machine fonts against clean backgrounds can be converted into ASCII using an OCR [1]. However, handwriting is much more difficult for OCRs to handle because of the wide variability present in handwriting (not only is there variability between writers, but a given person’s writing also varies).

Image matching of words has been used to recognize words in documents which use machine fonts [5, 10]. Recognition rates are much higher than when the OCR is used directly [10]. Machine fonts are simpler to match than handwritten fonts since the variation is much smaller; multiple instances of a given word printed in the same font are identical except for noise. In handwriting, however, multiple instances of the same word on the same page by the same writer show variations. The first two pictures in Figure 6 are two identical words from the same document, written by the same writer. It may thus be necessary to account for these variations.

3.2 Outline of Algorithm

1. A scanned greylevel image of the document is obtained.

2. The image is first reduced by half by gaussian filtering and subsampling.

3. The reduced image is then binarized by thresholding the image.

4. The binary image is now segmented into words. this is done by a process of smoothing and thresholding (see [18]).

5. A given word image (i.e. the image of a word) is used as a template. and matched against all the other word images. This is repeated for every word in the document. The matching is done in two phases. First, the number of words to be matched is pruned using the areas and aspect ratios of the word images - the word to be matched cannot have an area or aspect ratio which is too different from the template. Next, the actual matching is done by using a matching algorithm. Two different matching algorithms are tried here. One of them only accounts for translation shifts, while the other accounts for affine matches. The matching divides the word images into equivalence classes - each class presumably containing other instances of the same word.

6. Indexing is done as follows. For each equivalence class, the number of elements in it is counted. The top n equivalence classes are then determined from this list. The equivalence classes with the highest number of words (elements) are likely to be stopwords (i.e. conjunctions like ‘and’ , articles like ‘the’, and prepositions like ‘of’) and are therefore eliminated from further consideration. Let us assume that of the top n, m are left after the stopwords have been eliminated. The user then displays one member of each of these m equivalence classes and assigns their ASCII interpretation. These m words can now be indexed anywhere they appear in the document.

We will now discuss the matching techniques in detail.

3.3 Determination of Equivalence Classes

The list of words to be matched is first pruned using the areas and aspect ratios of the word images. The pruned list of words is then matched using a matching algorithm.

3.4 Pruning

It is assumed that

$$\frac{1}{\alpha} \leq \frac{A_{word}}{A_{template}} \leq \alpha \tag{1}$$

where $A_{template}$ is the area of the template and A_{word} is the area of the word to be matched. Typical values of α used in the experiments range between 1.2 and 1.3. A similar filtering step is performed using aspect ratios (ie. the width/height ratio). It is assumed that

$$\frac{1}{\beta} \leq \frac{Aspect_{word}}{Aspect_{template}} \leq \beta. \tag{2}$$

The value of β used in the experiments range between 1.4 and 1.7. In both the above equations, the exact factors are not important but it should not be so large so that valid words are omitted, nor so small so that too many words are passed onto the matching phase. The pruning values may be automatically determined by running statistics on samples of the document [15].

3.5 Matching

The template is then matched against the image of each word in the pruned list. The matching function must satisfy two criteria:

1. It must produce a low match error for words which are similar to the template.
2. It must produce a high match error for words which are dissimilar.

Two matching algorithms have been tried. The first algorithm - Euclidean Distance Mapping (EDM) - assumes that no distortions have occured except for relative translation and is fast. This algorithm usually ranks the matched words in the correct order (i.e. valid words first, followed by invalid words) when the variations in words is not too large. Although, it returns the lowest errors for words which are similar to the template, it also returns low errors for words which are dissimilar to the template. The second algorithm [28],referred to as SLH here, assumes an affine transformation between the words. It thus compensates for some of the variations in the words. This algorithm not only ranks the words in the correct order for all examples tried so far, it also seems to be able to better discriminate between valid words and invalid words. As currently implemented the SLH algorithm is much slower than the EDM algorithm (we expect to be able to speed it up).

3.6 Using Euclidean Distance Mapping for Matching

This approach is similar to that used by [6] to match machine generated fonts. A brief description of the method follows (more details are available from [18]).

Consider two images to be matched. There are three steps in the matching:

1. First the images are roughly aligned. In the vertical direction, this is done by aligning the baselines of the two images. In the horizontal direction, the images are aligned by making their left hand sides coincide.

   The alignment is, therefore, expected to be accurate in the vertical direction and not as good in the horizontal direction. This is borne out in practice.

2. Next the XOR image is computed. This is done by XOR'ing corresponding pixels (see Figure 6).

3. An Euclidean distance mapping [2] is computed from the XOR image by assigning to each white pixel in the image, its minimum distance to a black pixel. Thus a white pixel inside a blob is assigned a larger distance than an isolated white pixel. An error measure E_{EDM} can now be computed by adding up the distance measures for each pixel.

4. Although the approximate translation has been computed using step 1, this may not be accurate and may need to be fine-tuned. Thus steps (2) and (3) are repeated while sampling the translation space in both x and y. A minimum error measure E_{EDMmin} is computed over all the translation samples.

3.7 SLH Algorithm for Matching

The EDM algorithm does not discriminate well between good and bad matches. In addition, it fails when there is significant distortion in the words. This happens with the writing of Erasmus Hudson (Figure 7). Thus a matching algorithm which models some of the variation is needed. A second matching algorithm (SLH), which models the distortion as an affine transformations, was therefore tried (note that it is expected that the real variation is probably much more complex). An affine transform is a linear transformation between coordinate systems. In two dimensions, it is described by

$$\mathbf{r}' = \mathbf{A}\mathbf{r} + \mathbf{t} \tag{3}$$

where $\mathbf{t}$ is a 2-D vector describing the translation, $\mathbf{A}$ is a 2 by 2 matrix which captures the deformation, $\mathbf{r}'$ and $\mathbf{r}$ are the coordinates of corresponding points in the two images between which the affine transformation must be recovered. An affine transform allows for the following deformations - scaling in both directions, shear in both directions and rotation.

The algorithm chosen here is one proposed by Scott and Longuet-Higgins [28] (see [16]). The algorithm recovers the correspondence between two sets of points I and J under an affine transform.

The sets I and J are created as follows. Every white pixel in the first image is a member of the set I. Similarly, every white pixel in the second image is a member of set J. First, the centroids of the point sets are computed and the origins of the coordinate systems is set at the centroid. The SLH algorithm is then used to compute the correspondence between the point sets.

Given the (above) correspondence between point sets I and J, the affine transform $\mathbf{A}, \mathbf{t}$ can be determined by minimizing the following least mean squares criterion:

$$E_{SLH} = \sum_l (I_l - \mathbf{A}J_l - \mathbf{t})^2 \tag{4}$$

where I_l, J_l are the (x,y) coordinates of point I_l and J_l respectively.

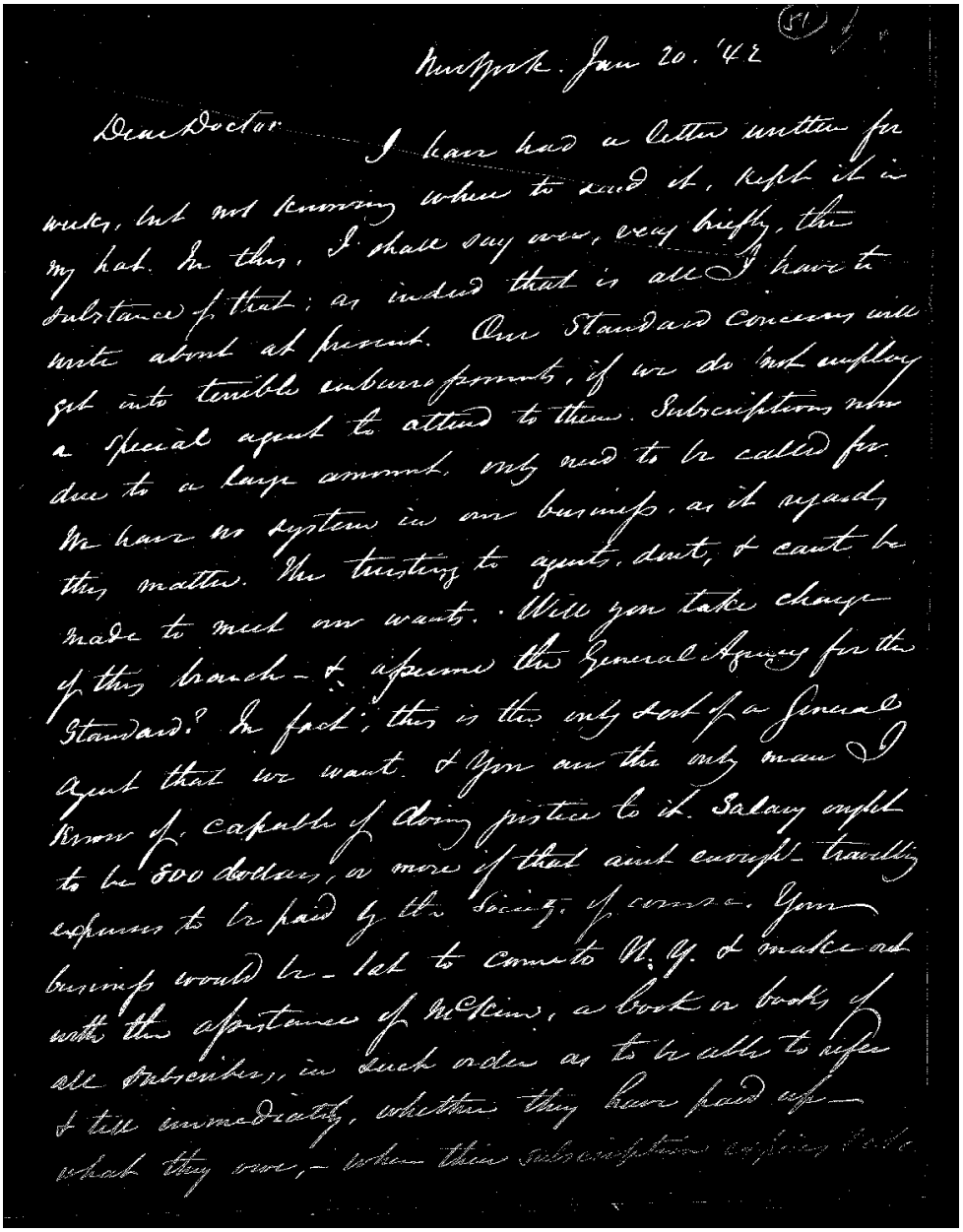
The values are then plugged back into the above equation to compute the error E_{SLH}. The error E_{SLH} is an estimate of how dissimilar two words are and the words can, therefore, be ranked according to it.

It will be assumed that the variation for valid words is not too large. This implies that if A_{11} and A_{22} are considerably different from 1, the word is probably not a valid match.

Note: The SLH algorithm assumes that pruning on the basis of the area and aspect ratio thresholds is performed.

3.8 Experiments

The two matching techniques were tested on two handwritten pages, each written by a different writer. The first page can be obtained from



New York. Jan 20. '42

Dear Doctor I have had a letter written for weeks, but not knowing where to send it, kept it in my hat. In this, I shall say now, very briefly, the substance of that; as indeed that is all I have to write about at present. Our Standard concerns will get into terrible embarrassments, if we do not employ a special agent to attend to them. Subscriptions now due to a large amount, only need to be called for. We have no system in our business, as it regards this matter. We trusting to agents, dont, & cant be made to meet our wants. Will you take charge of this branch - & assume the General Agency for the Standard? In fact; this is the only sort of a General Agent that we want. & You are the only man I know of, capable of doing justice to it. Salary ought to be 800 dollars, or more if that aint enough - travelling expenses to be paid by the Society, of course. Your business would be - 1st to come to N. Y. & make out with the assistance of McElrath, a book or books of all Subscribers, in such order as to be able to refer & tell immediately, whether they have paid up - what they owe, - when their subscriptions expire &c

the DIMUND document server on the internet http://documents.cfar.umd.edu/resources/database/handwriting.database.html This page will be referred to as the Senior document. The handwriting on this page is fairly neat (see [18] for a picture). The second page is from an actual archival collection - the Hudson collection from the library of the University of Massachusetts (part of the page is shown in Figure (7). This page is part of a letter written by James S. Gibbons to Erasmus Darwin Hudson. The handwriting on this page is difficult to read and the indexing technique helped in deciphering some of the words.

The experiments will show examples of how the matching techniques work on a few words. For more examples of the EDM technique see [18]. For more examples using the SLH technique and comparisons with the EDM technique see [16]. In general, the EDM method ranks most words in the Senior document correctly but ranks some words in the Hudson document incorrectly. The SLH technique performs well on both documents.

Both pages were segmented into words (see [18] for details) The algorithm was then run on the segmented words. In the following figures, the first word shown is the template. After the template, the other words are ranked according to the match error. Note that only the first few results of the matching are shown although *the template has been matched with every word on the page.* The area threshold α was chosen to be 1.2 and the aspect ratio threshold β was chosen as 1.4. The translation values were sampled to within ± 4 pixels in the X direction and ± 1 pixel in the y direction. Experimentally, this gave the best results.

3.9 Results using Euclidean Distance Mapping

The Euclidean Distance Mapping algorithm works reasonably well on the Senior document. An example is shown below.

In Figure (8), the template is the word "Lloyd". The figure shows that the four other instances of "Lloyd" present in the document are ranked before any of the other words. As Table (2) shows, the match errors for other instances of "Lloyd" is less than that for any other word. In the table, the first column is the Token number (this is needed for identification purposes), the second column is a transcription of the word, the third column shows the area in pixels, the fourth gives the match error and the last two columns specify the translation in the x and y directions respectively. Note the significant change in area of the words.

The performance on other words in the Senior document is comparable (for other examples see [18]). This is because the page is written fairly neatly. The performance of the method is expected to correlate with the quality of the handwriting. This was verified by running experiments on a page from the Hudson collection (Figure 7). The handwriting in the Hudson collection is difficult to read even for humans looking at grey-level images at 300 dpi The writing shows wide variations in size - for example, the area of the word "to" varies by as much as 100% ! However, this large a variation is not expected to occur and is not seen when the words are larger. Since humans have difficulty reading this material, we do not expect that the method will perform very well on this document.

Figure 8: Ranked matches for template "Lloyd" using the EDM algorithm (the rankings are ordered from left to right and from top to bottom).

The Euclidean Distance Mapping technique fails for the template "Standard" in the Hudson document (see Figure (9)). The failure occurs because the two instances of "Standard" are written differently. The template "Standard" has a gap between the "t" and the "a". This gap is not present in the second example of "Standard" (this is more clearly visible in Figure (10). A technique to model some distortions is, therefore, necessary.

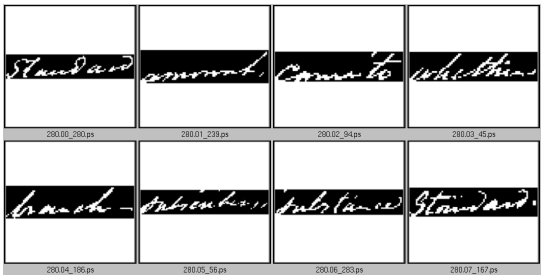


Figure 9: Rankings for template "Standard" using the EDM algorithm(the rankings are ordered from left to right and from top to bottom).

3.10 Experiments Using the SLH Algorithm

The SLH algorithm handles affine distortions and is, therefore more powerful then the EDM algorithm. Since

Token	Word	Area	E_{EDMmin}	Xshift	Yshift
105	Lloyd	1360	0.000	0	0
70	Lloyd	1224	0.174	0	0
165	Lloyd	1230	0.175	-2	0
197	Lloyd	1400	0.194	4	0
239	Lloyd	1320	0.197	-3	0
21	Maybe	1147	0.199	-1	0
180	along	1156	0.200	1	0
215	party	1209	0.202	1	0
245	spurt	1170	0.205	-1	0
121	dreary	1435	0.206	3	0

Table 2: Rankings and match Errors for template "Lloyd".

Token	Word	Area	CP	E_{SLH}	A		T
105	Lloyd	1368	233	0.00	1.00	0.00	0.00
					0.00	1.00	0.00
197	Lloyd	1400	199	1.302	0.96	-0.04	1.58
					0.01	1.04	0.14
70	Lloyd	1224	176	1.356	0.94	0.09	-1.02
					0.03	0.92	-1.38
165	Lloyd	1230	189	1.631	1.03	0.05	-0.43
					-0.01	0.87	-2.60
239	Lloyd	1320	203	1.795	0.99	-0.05	1.44
					0.03	1.07	2.21
157	lawyer	1518	185	3.393	0.96	-0.03	1.89
					0.05	1.11	0.03
240	Selwyn	1564	188	3.673	0.94	0.06	-4.23
					0.05	1.05	-0.75
91	thought	1178	181	3.973	0.97	0.03	2.33
					-0.01	1.08	2.91

Table 3: Rankings and Match Errors for template "Lloyd" Using SLH Algorithm.

the current version of the SLH algorithm is slow, the initial matches were pruned using the EDM algorithm and then the SLH algorithm run on the pruned subset.

Experiments were performed using both the Senior document and the Hudson documents. A few examples are shown here (for more details see [16]). For the Senior documents the same pruning ratios were chosen as before. To account for the large variations in the Hudson papers, the area threshold α was fixed at 1.3 and the aspect ratio threshold at 1.7. The value of σ depends on the expected translation. Since it is small, $\sigma = 2.0$. A lower value of $\sigma = 1.5$ yielded poorer results.

The matches for the template "Lloyd" are shown in Table (3). The succesive columns of the table, tabulate the Token Number, the transcription of the word, the area of the word image, the number of corresponding points recovered by the SLH algorithm, the match error E_SLH using the SLH algorithm and the affine transform. The entries are ranked according to the match error E_{SLH}. If either of A_{11} or A_{22} is less than 0.8 or greater than 1/0.8, that word is eliminated from the rankings. A comparison with Table (2) shows that the rankings change. This is not only true of the invalid words (for example the sixth entry in Table (2) is "Maybe" while the sixth entry in Table (3) is "lawyer" but is also true of the "Lloyd"'s. Both tables rank instances of "Lloyd" ahead of other words. The technique also shows a much greater discrimination in match error - the match error for "lawyer" is almost double the match error for the fifth "Lloyd".

The method was also run on the Hudson document (Figure (7)) and it ranked most of the words correctly on this document. As an example, we look at the word "Standard" on which the EDM method did not rank correctly. The SLH method produces the correct ranking inspite of the significant distortions in the word (see Figure (10)).

3.10.1 Recall–Precision Results

Indexing and retrieval techniques may be evaluated using recall and precision. Recall is defined as the "proportion of relevant documents actually retrieved" while precision is defined as the "proportion of retrieved documents that are relevant" [31]. Figure 3.10.1 shows the recall–precision results for both algorithms on the Senior

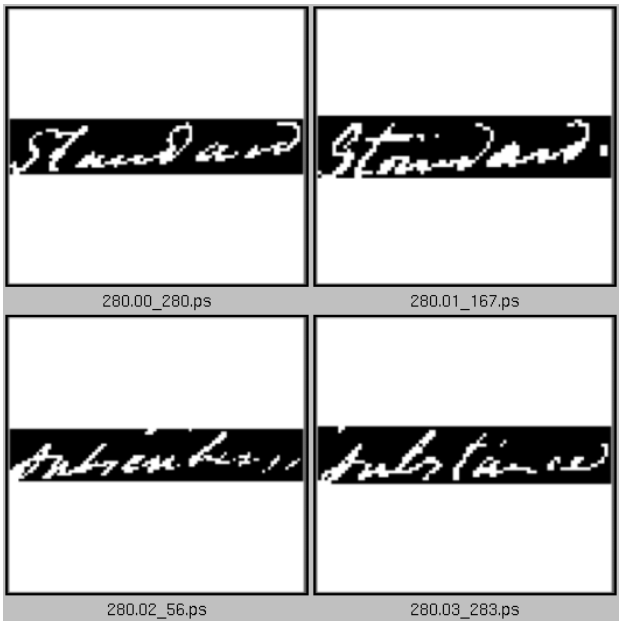


Figure 10: Rankings for template "Standard" for the SLH algorithm (the rankings are ordered from left to right and from top to bottom).

document. The two EDM graphs are for two different values of the area ratio (1.22 and 1.3). Notice that they do not differ significantly, thus showing that the exact values of the area ratio are not significant. The average precision for the EDM and SLH algorithms on the Senior document are 79.7 % and 86.3 % respectively. Note that SLH performs significantly better than EDM. Similar results are obtained with the Hudson document.

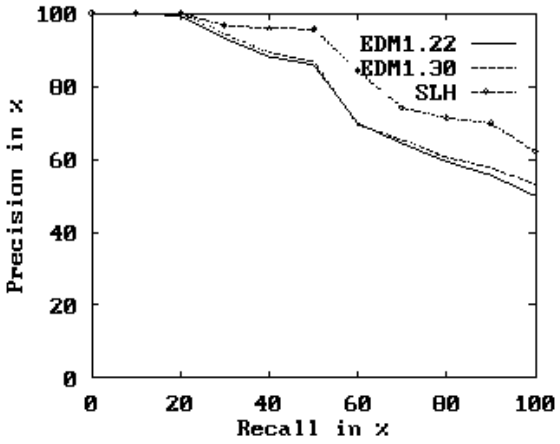


Figure 11: Recall precision results for Senior document

4 Image Retrieval

The indexing and retrieval of images using their content is a poorly understood and difficult problem. A person using an image retrieval system usually seeks to find semantic information. For example, a person may be looking for a picture of a leopard from a certain viewpoint. Or alternatively, the user may require a picture of Abraham Lincoln from a particular viewpoint.

Retrieving semantic information using image content is difficult to do. The automatic segmentation of an image into objects is a difficult and unsolved problem in computer vision. However, many image attributes like color, texture, shape and "appearance" are often directly correlated with the semantics of the problem. For example, logos or product packages (e.g., a box of Tide) have the same color wherever they are found. The coat of a leopard has a unique texture while Abraham Lincoln's appearance is uniquely defined. These image attributes can often be used to index and retrieve images.

The Center has carried out pioneering research in this area. The Center conducts research in both color based image retrieval see and appearance based image retrieval (the methods applied to appearance based image retrieval may also be directly applied to texture based image retrieval). We will now discuss appearance based retrieval (the reader is referred to [3] for discussions about the color based retrieval.

4.1 Retrieval by Appearance

Some attempts have been made to retrieve objects using their shape [4, 24]. For example, the QBIC system [4], developed by IBM, matches binary shapes. It requires that the database be segmented into objects. Since automatic segmentation is an unsolved problem, this requires the user to manually outline the objects in the database. Clearly this is not desirable or practical.

Except for certain special domains, all methods based on shape are likely to have the same problem. An object's appearance depends not only on its three dimensional shape, but also on the object's albedo, the viewpoint from which it is imaged and a number of other factors. It is non-trivial to separate the different factors constituting an object's appearance. For example, it is usually not possible to separate an object's three dimensional shape from the other factors.

The Center has overcome this difficulty by developing methods to retrieve objects using their appearance [26, 27, 19, 25]. The methods involve finding objects similar in appearance to an example object specified by the query.

To the best of our knowledge, ours is the first general query by appearance image retrieval system. Systems have been built to retrieve specific objects like faces (e.g., [29])). However, these systems require a number of training examples and it is not clear whether they can be generalized to retrieve other objects.

Some of the salient features of our system include:

1. The ability to retrieve "similar" images. This is in contrast with techniques which try to recover the *same* object. In our system, a car used as a query

will also retrieve other cars rather than retrieving only cars of a specific model.

2. The ability to retrieve images embedded in a background (see for example the cars in Figure 13 which appear against various backgrounds).
3. It does not require any prior manual segmentation of the database.
4. No training is required.
5. It can handle a range of variations in size.
6. It can handle 3D viewpoint changes up to about 20 to 25 degrees.

The user constructs the query by taking an example picture, and marking regions which she considers important aspects of the object. The query may be refined later depending on the retrieval results. Consider, for example, the first car shown in Figure 4.1. The user marks the region shown in the figure using a mouse. Notice that the region reflects the fact that wheels are central to a car. The user's query in this situation is to find visually similar objects (i.e., other cars) from a similar viewpoint (where the viewpoint can vary up to 25 degrees from the query).

The database images are filtered with derivatives of Gaussians at multiple scales. Derivatives of the first and second order are used. Differential invariants (invariants to 2D rotation) are created using the derivatives. [19, 25]. An inverted list is constructed from these invariants. The inverted list is indexed using the value of each invariant. The entire computation may be carried out off-line.

The on-line computation consists of calculating invariants for points in the query (which is a region in the image). Points with similar invariant values are now recovered from the database by indexing on the invariant values. The points obtained by indexing must also satisfy certain spatial constraints. That is, the values of the invariants at a pixel and at some of its neighbors must match. This ensures that the indexing scheme preserves the spatial layout of objects. Points which satisfy this spatial relationship vote and the database images are ranked on the basis of this vote.

The scheme described above works if the object is roughly the same size in the query and the image database. In practice it is quite common for the objects to be of different sizes in a database. The variation in size is handled by doing a search over scale space. That is, the query is filtered with Gaussian derivatives of different standard deviations [14, 13, 12] and the image simultaneously warped. This allows objects over a range of sizes to be matched [26, 27].

The query is outlined by the user with a mouse Figure 4.1. Figure 13 shows the results of a query. Notice that a large number of cars with white wheels have been retrieved. For more examples, see [19, 25]. This retrieval

Figure 12: Car Query for retrieval by indexing

was performed on a database of 1600 images taken from the Internet, the Library of Congress and other sources. The database consists of faces, monkeys, apes, cars, diesel and steam locomotives and a few houses. Lighting and camera parameters are not known.

5 Conclusion

This paper has described the multimedia indexing and retrieval work being done at the Center for Intelligent Information Retrieval. Work on systems for finding text in images, indexing archival handwritten documents and image retrieval by content has been described. The research described is part of an on-going research effort focused on indexing and retrieving multimedia information in as many ways as possible. The work described here has many applications, principally in the creation of the digital libraries of the future.

6 Acknowledgements

This paper includes research contributions by Victor Wu and Srinivas Ravela of the multimedia indexing and retrieval group. David Hirvonen and Adam Jenkins provided programming support. Ed Riseman gave comments on some of this work. I would like to thank Bruce Croft and CIIR for supporting this work and Gail Giroux and the University of Massachusetts Library for the scanned page from the Hudson collection.

References

[1] M. Bokser. Omnidocument technologies. *Proceedings IEEE*, 80(7):1066–1078, 1992.

[2] Per-Erik Danielsson. Euclidean distance mapping. *Computer Graphics and Image Processing*, 14:227–248, 1980.

[3] M. Das, E. M. Riseman, and B. A. Draper. Focus : Searching for multi-colored objects in a diverse image database. *accepted to the IEEE CVPR '97*, June 1997.

[4] Myron Flickner et al. Query by image and video content: The qbic system. *IEEE Computer Magazine*, pages 23–30, Sept. 1995.

[5] L. D. Wilcox F. R. Chen, D. S. Bloomberg. Spotting phrases in lines of imaged text. In *Proceedings of the SPIE conf. on Document Recognition II*, volume 2422, pages 256–269, San Jose, CA, Feb. 1995.

[6] Paul Filiski and Jonathan J. Hull. Keyword selection from word recognition results using definitional overlap. In *Third Annual Symposium on Document Analysis and Information Retrieval, UNLV, Las Vegas*, pages 151–160, 1994.

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

[8] Anil K. Jain and Sushil Bhattacharjee. Text Segmentation Using Gabor Filters for Automatic Document Processing. *Machine Vision and Applications*, 5, 1992.

[9] G. J. F. Jones, J. T. Foote, K. Sparck Jones, and S. J. Young. Video mail retrieval: The effect of word spotting accuracy on precision. In *International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 309–316, 1995.

[10] Siamak Khoubyari and Jonathan J.Hull. Keyword location in noisy document image. In *Second Annual Symposium on Document Analysis and Information Retrieval, UNLV, Las Vegas*, pages 217–231, 1993.

[11] J. Malik and P. Perona. Preattentive texture discrimination with early vision mechanisms. *Journal of the Optical Society of America A*, 7(5):923–932, May 1990.

[12] R. Manmatha. Image matching under affine deformations. In *Invited Paper, Proc. of the 27nd Asilomar IEEE Conf. on Signals, Systems and Computers*, pages 106–110, 1993.

[13] R. Manmatha. A framework for recovering affine transforms using points, lines or image brightnesses. In *Proc. Computer Vision and Pattern Recognition Conference*, pages 141–146, 1994.

[14] R. Manmatha. Measuring the affine transform using gaussian filters. In *Proc. 3rd European Conference on Computer Vision*, pages 159–164, 1994.

[15] R. Manmatha and W. B. Croft. Word spotting: Indexing handwritten manuscripts. In Mark Maybury, editor, *Intelligent Multi-media Information Retrieval*. AAAI/MIT Press, April 1998.

[16] R. Manmatha, Chengfeng Han, and E. M. Riseman. Word spotting: A new approach to indexing handwriting. Technical Report CS-UM-95-105, Computer Science Dept, University of Massachusetts at Amherst, MA, 1995.

[17] R. Manmatha, Chengfeng Han, and E. M. Riseman. Word spotting: A new approach to indexing handwriting. In *Proc. Computer Vision and Pattern Recognition Conference*, pages 631–637, 1996.

[18] R. Manmatha, Chengfeng Han, E. M. Riseman, and W. B. Croft. Indexing handwriting using word matching. In *Digital Libraries '96: 1st ACM International Conference on Digital Libraries*, pages 151–159, 1996.

[19] R. Manmatha and S. Ravela. A syntactic characterization of appearance and its application to image retrieval. In *Proceedings of the SPIE conf. on Human Vision and Electronic Imaging II*, volume 3016, San Jose, CA, Feb. 1997.

[20] S. Mori, C. Y. Suen, and K. Yamamoto. Historical review of ocr research and development. *Proceedings of the IEEE*, 80(7):1029–1058, July 1992.

[21] G. Nagy, S. Seth, and M. Viswanathan. A Prototype Document Image Analysis System for Technical Journals. *Computer*, pages 10–22, July 1992.

[22] Lawrence O'Gorman. The Document Spectrum for Page Layout Analysis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(11):1162–1173, Nov. 1993.

[23] Theo Pavlidis and Jiangying Zhou. Page Segmentation and Classification. *CVGIP: Graphical Models and Image Processing*, 54(6):484–496, Nov. 1992.

[24] A. Pentland, R. W. Picard, and S. Sclaroff. Photobook: Tools for content-based manipulation of databases. In *Proc. Storage and Retrieval for Image and Video Databases II,SPIE*, volume 185, pages 34–47, 1994.

[25] S. Ravela and R. Manmatha. Image retrieval by appearance. In *Accepted to the 20th Intl. Conf. on Research and Development in Information Retrieval (SIGIR'97)*, July 1997.

[26] S. Ravela, R. Manmatha, and E. M. Riseman. Image retrieval using scale-space matching. In Bernard Buxton and Roberto Cipolla, editors, *Computer Vision - ECCV '96*, volume 1 of *Lecture Notes in Computer Science*, Cambridge, U.K., April 1996. 4th European Conf. Computer Vision, Springer.

[27] S. Ravela, R. Manmatha, and E. M. Riseman. Scale space matching and image retrieval. In *Proc. DARPA Image Understanding Workshop*, 1996.

[28] G. L. Scott and H. C. Longuet-Higgins. An algorithm for associating the features of two patterns. *Proc. Royal Society of London B*, B244:21–26, 1991.

[29] M. Turk and A. Pentland. Eigenfaces for recognition. *J. of Cognitive NeuroScience*, 3:71–86, 1991.

[30] H.R. Turtle and W.B. Croft. A comparison of text retrieval models. *Computer Journal*, 35(3):279–290, 1992.

[31] C. J. van Rijsbegen. *Information Retrieval*. Butterworths, 1979.

[32] F. Wahl, K. Wong, and R. Casey. Block segmentation and text extraction in mixed text/image documents. *Computer Vision Graphics and Image Processing*, 20:375–390, 1982.

[33] D. Wang and S. N. Srihari. Classification of newspaper image blocks using texture analysis. *Computer Vision Graphics and Image Processing*, 47:327–352, 1989.

[34] V. Wu, R. Manmatha, and E. M. Riseman. Finding Text In Images. *Technicial Report 97-09, Computer Science Department, UMass, Amherst, MA*, 1997.

[35] V. Wu, R. Manmatha, and E. M. Riseman. Finding Text In Images. *accepted to the Second ACM Intl. conf. on Digitial Libraries DL'97*, July 1997.

Figure 13: The results of the car query.