TextFinder: An Automatic System to Detect and Recognize Text In Images

Victor Wu, Raghavan Manmatha, *Member, IEEE*, and Edward M. Riseman, *Sr. Member, IEEE*

Abstract—A robust system is proposed to automatically detect and extract text in images from different sources, including video, newspapers, advertisements, stock certificates, photographs, and checks. Text is first detected using multiscale texture segmentation and spatial cohesion constraints, then cleaned up and extracted using a histogram-based binarization algorithm. An automatic performance evaluation scheme is also proposed.

Index Terms—Text reading, character recognition, multimedia indexing, text detection, texture segmentation, filters, hierarchical processing, binarization, connected-components.

1 Introduction

MOST information available today is either on paper or in the form of photographs and videos. To build digital libraries, this information needs to be digitized into images and the text converted to ASCII for storage, retrieval, and manipulation. However, current optical character recognition (OCR) technology [1], [8] is restricted to finding text printed against clean backgrounds, and cannot handle text printed against shaded or textured backgrounds or embedded in images. More sophisticated text reading systems employ document analysis (page segmentation) schemes to identify text regions before applying OCR so that time is not spent trying to interpret nontext items. Etemad et al. [2] used a neural net to classify the output of wavelets into text and nontext regions. The neural net requires a rich set of training examples to work effectively. Other schemes require clean binary input [3], [13], [14], [15]; some assume specific document layouts such as video frames [12], newspapers [6], technical journals [9], or are domain specific like mail address blocks [11]. Thus, there is a need for robust systems which extract and recognize text from general backgrounds.

2 A New System

The system takes advantage of the distinctive characteristics of text which make it stand out from other image material. For example, by looking at the comic page of a newspaper a few feet away, one can probably tell quickly where the text is without actually recognizing individual characters. Intuitively, text has the following distinguishing characteristics: 1) text possesses certain frequency and orientation information; 2) text shows **spatial cohesion**—characters of the same text string (a word, or words in the same line) are of similar heights, orientation, and spacing.

2.1 Step 1: Texture Segmentation Module

The first characteristic suggests that text may be treated as a distinctive texture. The first phase of the system therefore, adapts a standard **Texture Segmentation** scheme (Fig. 1) to segment text regions. This consists of a linear filtering stage followed by a nonlinear stage [7]. Nine second order derivatives of Gaussians at scales of $\sigma = \{1, \sqrt{2}, 2\}$ are used. Each filter output is passed through the nonlinear transformation $tanh(\alpha t)$, where $\alpha = 0.25$. For each pixel location, local energy estimates are computed using the outputs of the nonlinear transformation, which form a feature vector for that pixel. The set of feature vectors is clustered using a K means algorithm (with K = 3). Fig. 2b shows results of texture segmentation applied to Fig. 2a.

The texture segmentation scheme used is not sufficient for text detection and extraction if images more complicated than clean newspaper scans have to be dealt with. Nevertheless, the segmentation result can be used as a focus of attention for further processing called **Chip Generation**.

2.2 Step 2: Chip Generation Module

The basic idea for chip generation is to apply a set of appropriate heuristics to find text strings within/near the segmented regions. The heuristics are designed to reflect the spatial cohesion of the text. The algorithm uses a bottom-up approach: significant edges form **strokes** (connected components); strokes are aggregated to form **chips** (regions) corresponding to text strings. The rectangular bounding boxes of the chips are used to indicate the locations of the hypothesized (detected) text strings. Chip Generation consists of the following ordered steps:

1. **Stroke Generation** by grouping edge pixels using connected components.
2. **Stroke Filtering** to eliminate strokes unlikely to belong to any horizontal text string.
3. **Stroke Aggregation** to form chips of connected strokes likely to belong to the same text string.
4. **Chip Filtering** to eliminate chips unlikely to correspond to horizontal text strings.
5. **Chip Extension** where filtered chip are treated as strokes and aggregated again to form chips which cover the text strings more completely.

The purpose of Stroke Filtering is to eliminate the false positive strokes with heuristics to capture the fact that neighboring characters/words in the same text string usually have similar heights and are horizontally aligned. It is reasonable to assume that similarity of character heights causes heights of corresponding stokes to be similar. These heuristics can be described using **connectability** defined as:

Definition. *Strokes A and B are* ***connectable*** *if they are of similar height and horizontally aligned, and there is a* ***path*** *between A and B, where a path is a horizontal sequence of consecutive pixels in the segmented region which connects A and B by 4-neighbor adjacency.*

Here, two strokes are considered to be of similar height if the height of a shorter stroke is at least 40 percent of that of a taller one. To determine the horizontal alignment, strokes are projected onto the Y-axis. If the overlap of the projections of two strokes is at least 50 percent of the shorter stroke, they are considered to be horizontally aligned. More formally, a stroke is eliminated if either 1) it does not sufficiently overlap with the segmented text regions. or 2) it has no connectable stroke. Condition 1 says the strokes are expected to overlap the segmented regions. Since text segmentation is not perfect, one cannot expect total overlap. A minimum of 30 percent overlap rate worked well for all the test images. Condition 2 says that if no path leads to some connectable stroke(s), it is probably an isolated stroke or line which does not belong to any text string.

Since characters which belong to the same text string are expected to be of similar height and horizontally aligned, the concept of connectability can be used to aggregate strokes to generate chips that correspond to text strings. By empirical

• *The authors are with the Department of Computer Science, University of Massachusetts, Amherst, MA 01003.*
E-mail: vwu@cs.umass.edu.

Manuscript received 27 Dec. 1998; revised 18 Aug. 1999.
Recommended for acceptance by R. Kasturi.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 108969.

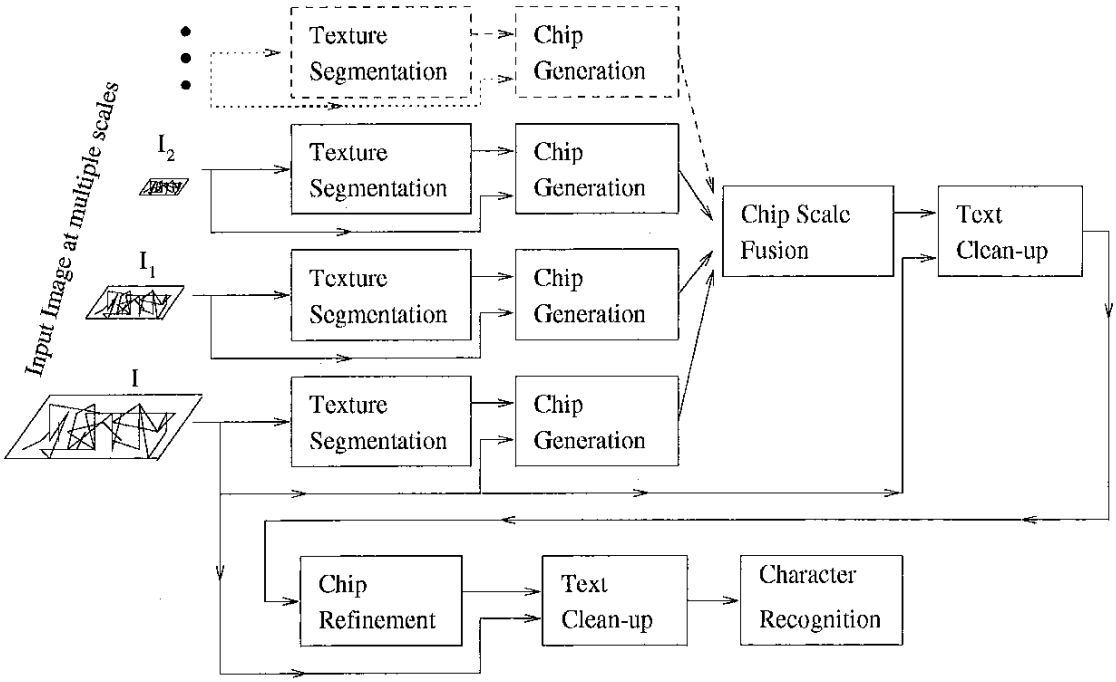


Fig. 1. The top level components of the text detection and extraction system. The pyramid of the input image is shown as $I, I_1, I_2...$

observation across a range of text sources, the spacing between the characters and words of a text string is usually less than twice the height of the tallest character, and so is the width of a character in most fonts. Therefore, the following criterion is used to generate chips: two strokes, A and B, are connected if they are connectable and there is a path between A and B whose length is less than twice the height of the taller stroke.

Text strings are expected to have a certain height in order to be reliably recognized by an OCR system. Thus, one choice is to filter out chips whose height is small. Furthermore, since we are

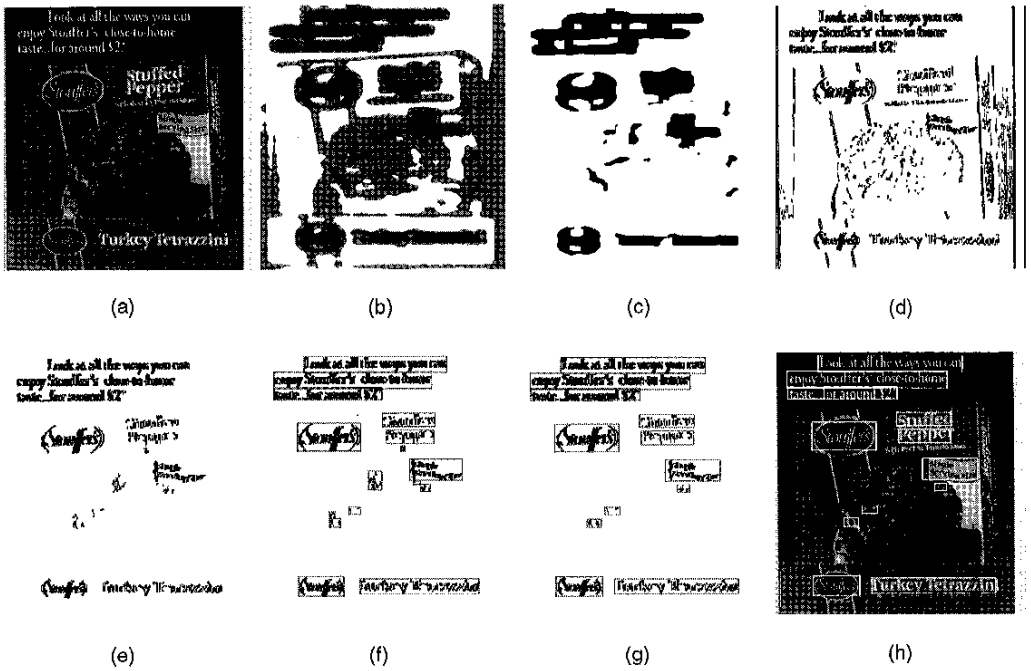


Fig. 2. An example of Texture Segmentation and Chip Generation at the half resolution level. (a) portion of an input image; (b) output of the clustering stage. Black regions are labeled as "text" regions; (c) the Text regions after the morphological closure operation; (d) strokes produced by Stroke Generation procedure; (e) filtered strokes; (f) chips (grey boxes) produced by applying Stroke Aggregation; (g) chips after Chip Filtering and Extension processes; (h) chips mapped to input image.

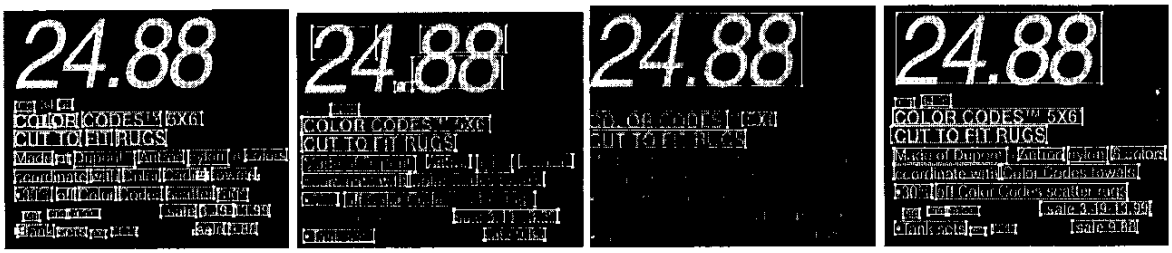


Fig. 3. The scale problem and its solution. (a) Chips generated for the input image at full resolution; (b) half resolution; (c) $\frac{1}{4}$ resolution; (d) chips generated at all three levels mapped onto the input image. Scale-redundant chips are removed. Notice that there is an optimum resolution for each range of font sizes.

interested in text strings, not just isolated characters, the width of a chip is also used to filter out text. Lastly, for horizontally aligned text strings, their aspect ratio (height/width) is usually large. Therefore, chips are filtered using the following constraints on their minimum bounding boxes: A chip is eliminated if the width of its box is less than cw_τ; or the height of its box is less than ch_τ or the aspect ratio (width/height) of its box is larger than $ratio_\tau$.

It is usually difficult even for a human to read the text when its height is less than seven pixels, thus 7 has been used for ch_τ for the experiments. A horizontal text string is usually longer horizontally, hence setting cw_τ to at least twice the minimum height seems reasonable. Thus, in all of our experiments, $cw_\tau = 15$ and $ch_\tau = 7$ were used. Normally, the width of a text string should be larger than its height. But in some fonts, the height of a character is larger than its width. Therefore, $ratio_\tau = 0.9$ is used here, attempting to cover that case to some extent.

Some of the strokes may only cover fragments of the corresponding characters. Therefore, these strokes might violate the constraints used for stroke filtering, and hence be eliminated. Consequently, some chips may only cover part of the corresponding text strings. Fortunately, this fragmentation problem can usually be corrected. Notice that the chips corresponding to the same text stroke are still horizontally aligned and of similar height. Thus, by treating the chips as strokes, the Stroke Aggregation procedure can be applied again to aggregate the chips into larger chips and capture more complete words in the extended chips.

2.3 Step 3: Chip Scale Fusion Module

The text detection procedures just outlined work well for text over a certain range of font sizes. To detect text whose font size varies significantly, the input image is processed at different resolutions. The output chip boxes generated at each resolution level are then mapped back onto the original image (**Chip Scale Fusion**). Fig. 3 shows an example of how chips at different scales are detected, with a hierarchy of three levels to find text of fonts up to 160 pixels in height (Fig.1).

2.4 Step 4: Text Cleanup Module

Since each of the generated text bounding boxes usually contains text of similar intensities and background (usually around single words or group of words), a single threshold suffices to clean up and binarize the corresponding region so text stands out. A simple, effective histogram-based algorithm, as described in [16], is used to find the threshold value automatically for each text region. This algorithm is used for the **Text Clean-up** module in the system.

2.5 Step 5: Chip Refinement Module

Nontext items might survive the previous processing and occur in the binarized output (Fig. 4b). Thus, a **Chip Refinement** phase is used in the system to filter them out. This is done by treating the

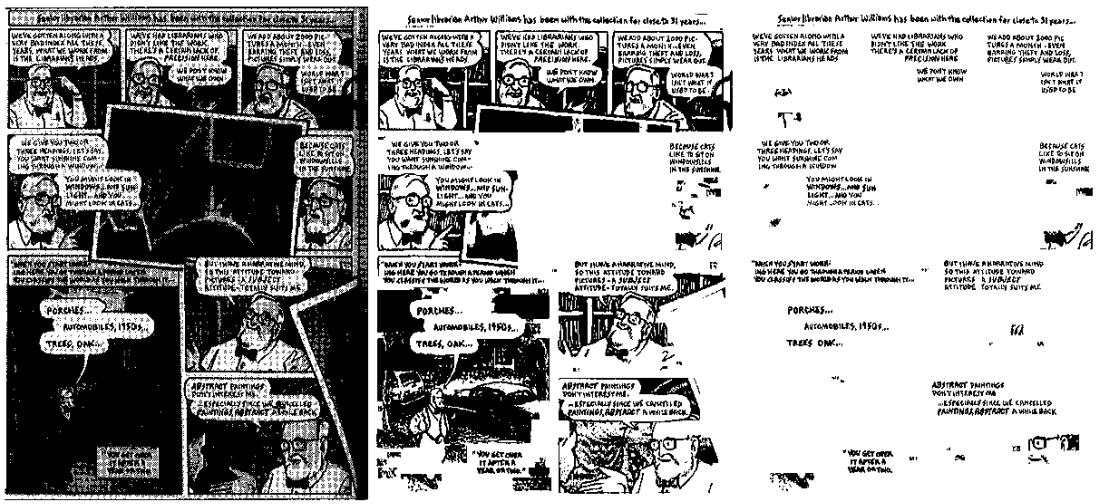


Fig. 4. (a) An input image from the New Yorker magazine. (b) Binarization result of **(a)** before the refinement step. (c) Binarization result of Fig. **(a)** after the refinement step.

TABLE 1
Distribution of the Image Type

Ads	Photo / Video	Check	Envelope	Map	Cartoon	Invoice	Other
20	8	3	1	1	5	2	8

extracted items (text and nontext) as strokes to regenerate chips using the same algorithms, with tighter constraints than those used in the Chip Generation phase (see [18] for more details). The chips produced this time usually enclose the text strings better. The Chip Clean-up process is then applied to the new chips to obtain better binarization results (Fig. 4).

Fig. 1 depicts the system described above. Experimental results have shown that the system works well with both machine generated fonts and some script fonts. Generally the system is not sensitive to the font sizes. The system is also stable and robust—all the system parameters remain constant for all of the text images from a wide variety of sources including newspapers, magazines, printed advertisement, photographs, and checks. Notice that some of these documents have structured layout, some do not, and the system works well in either case. A detailed description of the design and implementation of the system is presented in [18].

3 Experiments

The database contains 48 images of which 27 are color images and the remaining 21 are grayscale images. The sizes of the images varied from 186 pixels by 349 pixels to 3,391 by 2,486 pixels. Tables 1 and 2 show the characteristics of the test images in the database. Sources of test images include internet web images, the Library of Congress, and other locally scanned documents. These test images came from a wide variety of sources: digitized video frames, photographs, newspapers, advertisements in magazines or sales flyers, personal checks, and envelopes. Some of the images have regular page layouts, others do not. No assumptions are made about resolution of input images, since such information is usually not available. All color documents were scanned as color images, and then converted into grayscale.

3.1 Evaluation

There has been little work on techniques to automatically evaluate text detection on such diverse document collections. Although two recent papers [12], [4] provide results on detection accuracy and false alarms, the methodology has either been restricted to specific classes of images or is not extendable for automatic evaluation on large collections of diverse images.

We propose a technique for automatic evaluation for text detection in images. First, ground truth is created by manually constructing minimum bounding boxes around each character in the image. The output produced by the text detection algorithm is compared against the ground truth. The percentage of the total number of characters that have been detected will be used as a measure of the detection accuracy of the sytem. A character is considered to be detected if it is completely covered by a generated text box. We test this approach to automatic evaluation by visual inspection by tabulating detected characters. We obtain comparable results using both the automatic evaluation and visual inspection.

The third column in Table 3 shows the results of manual counting, while the fourth column shows the results obtained using the automatic evaluation scheme. The system does well in detecting text whose font height is more than 10 pixels, but as expected it does not work well for very small characters. Comparing columns 3 and 4 of Table 3, it is clear that performance of automatic evaluation is close (but slightly lower) to that of visual inspection (see [18] for details). For the test image set, the false alarm rate was 5.6 percent. The false alarm rate is defined here as the total area covered by the false alarm pixels as a percentage of the image size. The accuracy of the cleanup-algorithm was also measured with further discussion and details appearing in [17], [16], [18].

The system was also tested by using an OCR to recognize the characters. For this experiment, a binary image is formed using all the cleaned-up text chips for each input image. Then these binary images are manually fed to to the OCR (Caere's OmniPage Pro 8.0) for recognition. This test also provides an objective evaluation of the text clean-up and binarization algorithm. Table 4 shows the results of the OCR test. The first column is a count of the machine printed characters and words in the database (since the OCR can only handle machine print), the second column lists the number extracted by the text detection system while the last column lists the recognition rate of the OCR. Note that for many of the input images, applying the OCR engine directly without the clean-up procedure yields very poor results, showsing the importance of robust binarization and clean-up.

Fig. 5a is the full original image of an advertisement for "Stouffer's" which has no structured layout. The final binarization result is shown in Fig. 5b. The corresponding OCR output (using Caere's OmniPage Pro 8.0) is shown in Fig. 5c. This example provides a feel for the overall performance of the system by showing whole images. In this format some fine details are lost due to the scaling of the images to fit the page. For example, words of smaller fonts and the word "Stouffer's" in script appear to be fragmented, although actually they are not. All the characters under "Stuffed Pepper" were missed due to their poor contrast and small size. The words are actually blurred, hence the region has very low energy. Notice that most of the texture in the picture of the food is filtered out, showing the robustness of the system.

The OCR engine correctly recognized much of the text of machine-printed fonts as shown in Fig. 5(c). It made mistakes on the "Stouffer's" trademarks since they are in script. In many places although the OCR made errors, the text was extracted acurrately.

TABLE 2
Font Size in Pixels

	min	max	mean	std. dev.
Height	6	197	25.9	13.9
Width	2	240	17.6	11.8

TABLE 3
Detection Rates

Height (pixels)	# of chars.	Chars detected by visual inspection	Chars detected automatically
<= 10	705	55.7%	55.2%
11 – 20	7571	91.0%	90.0%
> 20	13754	96.8%	95.2%
>= 6	22030	93.5%	92.1%

TABLE 4
OCR Test Results

	Total in database	Extracted	Recognized		
			Count	% over Extracted	% over Total
Char	18688	16664	15656	94.0%	83.8%
Word	4042	3304	2926	88.6%	72.4%

The system's speed is a function of the size of the original image. For an image of size 320 pixels by 240 pixels it takes 10 s, while for an image of size 1,512 pixels by 1,517 pixels it takes 291 s (all times are for a 200 Mhz Pentium Pro PC with 128 Mbytes of memory). Most of the time is spent in the texture segmentation phase, which can be significantly shortened by using just one frequency channel (instead of 3) corresponding to $\sigma = \sqrt{2}$. With one frequency channel, the times above are reduced to 3 s and 143 s, respectively.

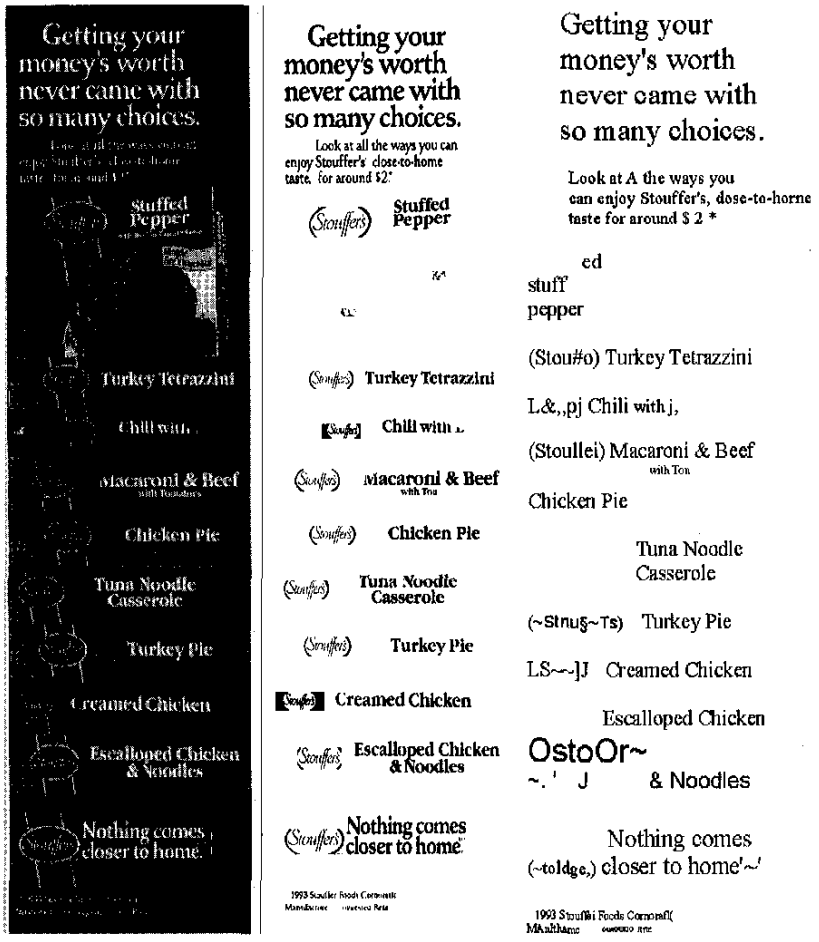
4 Discussion

Although there are systems [19], [5], [4] which utilize color information to detect text in color images, the system described in this paper does not use color. Our system starts by extracting significant intensity edges (strokes). Nevatia [10] pointed out that in general color edges are also intensity edges. Thus, "strokes" which were originally in color are usually still present in the converted grayscale image. It is possible that some of the characters may be printed in colors that have little monochrome contrast with the background, thereby causing our system difficulty. However, we believe (and our results show) that such characters are not common.

The system presented is not sensitive to image resolution. It works particularly well in extracting text from textured and/or hatched background. However, it tends to have problems extracting very small text (font height less than 10 pixels) or text with poor contrast, as all systems do. Reasons for this include the difficulty of extracting strokes under these circumstances and also misclassification by the texture segmentation phase when the contrast is poor.

5 Conclusion

Current OCR and other document segmentation and recognition technologies do not work well for documents with text printed against shaded or textured backgrounds or those with nonstructured layouts. In contrast, we have proposed a text extraction system which works well for normal documents as well as documents described in the above situations. The



(a) (b) (c)

Fig. 5. Example 1. (a) Original image; (b) extracted text; (c) OCR result.

system is stable and robust, with all the system parameters remaining constant throughout all the experiments on a diverse test set. The system detected 92 percent of characters with a height greater than 10 pixels.

ACKNOWLEDGMENTS

We would like to thank Bruce Croft and CIIR for supporting this work. Adam Jenkins helped port the code to a standalone version while Jonathan Lim provided system support. We would also like to thank Allen Hanson, and Yong-Qing Cheng for their constructive comments and suggestions.

This material is based on work supported in part by the U.S. National Science Foundation, Library of Congress and Department of Commerce under cooperative agreement number EEC-9209623, in part by the United States Patent and Trademark Office and Defense Advanced Research Projects Agency/ITO under ARPA order number D468, issued by ESC/AXS contract number F19628-95-C-0235, in part by U.S. the Air Force Office of Scientific Research under grant number F49620-99-1-0138, in part by the U.S. National Science Foundation under grant number IRI-9619117 and in part by U.S. National Science Foundation 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.

REFERENCES

[1] M. Bokser, "Omnidocument Technologies," *Proc. IEEE*, vol. 80, no. 7, pp. 1,066-1,078, July 1992.
[2] K. Etemad, D. Doermann, and R. Chellapa, "Multiscale Segmentation of Unstructured Document Pages Using Soft Decision Integration," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 1, pp. 92-96, Jan. 1997.
[3] L.A. Fletcher and R. Kasturi, "A Robust Algorithm for Text String Separation from Mixed Text/Graphics Images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 10, no. 6, pp. 910-918, Nov. 1988.
[4] A.K. Jain and B. Yu, "Automatic Text Location in Images and Video Frames," *Pattern Recognition*, vol. 31, no. 12, pp. 2,055-2076, 1998.
[5] D.L.J. Zhou and T. Tasdizen, "Extracting Text from WWW Images," *Proc. SPIE'98 Document Recognition V*, pp. 130-138, Jan. 1998.
[6] M. Kamel and A. Zhao, "Extraction of Binary Character/Graphics Images from Grayscale Document Images," *Computer Vision, Graphics, and Imaging Processing*, vol. 55, no. 3, pp. 203-217, May 1993.
[7] J. Malik and P. Perona, "Preattentive Texture Discrimination with Early Vision Mechanisms," *J. Opt. Soc. Am.*, vol. 7, no. 5, pp. 923-932, May 1990.
[8] S. Mori, C.Y. Suen, and K. Yamamoto, "Historical Review of OCR Research and Development," *Proc. IEEE*, vol. 80, no. 7, pp. 1,029-1,058, July 1992.
[9] G. Nagy, S. Seth, and M. Viswanathan, "A Prototype Document Image Analysis System for Technical Journals," *Computer*, pp. 10-22, July 1992.
[10] R. Nevatia, "A Color Edge Detector and Its Use in Scene Segmentation" *IEEE Trans. System, Man, and Cybernetics*, vol. 7, no. 11, pp. 820-826, Nov. 1977.
[11] P.W. Palumbo, S.N. Srihari, J. Soh, R. Sridhar, and V. Demjanenko, "Postal Address Block Location in Real Time," *Computer*, pp. 34-42, July 1992.
[12] M.A. Smith and T. Kanade, "Video Skimming and Characterization through the Combination of Image and Language Understanding Techniques," *Proc. IEEE Computer Vision and Pattern Recognition '97*, pp. 775-781, 1997.
[13] F.M. Wahl, K.Y. Wong, and R.G. Casey, "Block Segmentation and Text Extraction in Mixed Text/Image Documents," *Computer Graphics and Image Processing*, vol. 20, pp. 375-390, 1982.
[14] D. Wang and S.N. Srihari, "Classification of Newspaper Image Blocks Using Texture Analysis," *Computer Vision, Graphics, and Imaging Processing*, vol. 47, pp. 327-352, 1989.
[15] K.Y. Wong, R.G. Casey, and F.M. Wahl, "Document Analysis System," *IBM Journal Res. Dev.*, vol. 26, no. 6, pp. 647-656, 1982.
[16] V. Wu and R. Manmatha, "Document Image Clean-Up and Binarization" *Proc. SPIE'98 Document Recognition V*, pp. 263-273, Jan. 1998.
[17] V. Wu, R. Manmatha, and E.M. Riseman, "Finding Text In Images," *Proc. the Second Int'l Conf. Digital Libraries*, pp. 1-10, Philadaphia, PA, July 1997.
[18] V. Wu, R. Manmatha, and E.M. Riseman, "TextFinder: An Automatic System to Detect and Recognize Text in Images," Technicial Report 99-40, Computer Science Dept., Univ. of Massachusetts, Amherst, 1999.
[19] Y. Zhong, K. Karu, and A.K. Jain, "Locating Text in Complex Color Images" *Pattern Recognition*, vol. 28, no. 10, pp. 1,523-1,536, Oct. 1995.

RANSAC-Based DARCES: A New Approach to Fast Automatic Registration of Partially Overlapping Range Images

Chu-Song Chen, *Member, IEEE Computer Society*,
Yi-Ping Hung, *Member, IEEE Computer Society*, and
Jen-Bo Cheng

Abstract—In this paper, we propose a new method, the RANSAC-based DARCES method, which can solve the partially overlapping 3D registration problem without any initial estimation. For the noiseless case, the basic algorithm of our method can guarantee that the solution it finds is the true one, and its time complexity can be shown to be relatively low. An extra characteristic is that our method can be used even for the case that there are no local features in the 3D data sets.

Index Terms—Computer vision, range data, range image, registration, 3D imaging.

1 INTRODUCTION

REGISTRATION of two partially overlapping range images taken from different views is an important task in 3D computer vision. In the past, a popular type of approach to solving the 3D registration problem has been the *iterative approach* [1], [6]. However, the drawbacks are that 1) they require a good initial estimate to prevent the iterative process from being trapped in a local minimum and 2) there is no guarantee of getting the correct solution even for the noiseless case. Many approaches have modified the two approaches proposed in [1] and [6] to obtain more reliable correspondence in each iteration [8], [11].

Another popular type of method is the *feature-based approach* [10], [7], [9]. Feature-based approaches have the advantage that they do not require initial estimates of the rigid-motion parameters. Their drawbacks are mainly that 1) they can not solve the problem in which the 3D data sets contain no prominent/salient local features and 2) a large percentage of the computation time is usually spent on preprocessing, which includes extraction of invariant features [7], [10] and organization of the extracted feature-primitives (e.g., sorting [7]). In addition, Blais and Levine expressed the 3D registration task as an optimization problem [2]. The very-fast-simulated-reannealing technique was used to find the global minimum of the error function.

Our goal in this paper is to solve the 3D registration problem in a fast and reliable manner. We propose a new method—the *data-aligned rigidity-constrained exhaustive search* (DARCES), which can check all possible data-alignments of two given 3D data sets in an efficient way while requiring no preprocessing and no initial estimates of the 3D rigid-motion parameters. Furthermore, to solve the partially overlapping 3D registration problem, the *random sample consensus* (RANSAC) scheme is integrated into the DARCES procedure.

• *The authors are with the Institute of Information Science, Academia Sinica, Nankang, Taipei, Taiwan.*
E-mail: {song, hung}@iis.sinica.edu.tw.

Manuscript received 1 June 1998; revised 9 Mar. 1999.
Recommended for acceptance by G. Medioni.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 107684.